```
pip install sdv
```

```
Collecting nvidia-cufft-cu12==11.0.2.54 (from torch>=1.11.0->ctgan>=0.10.0->sdv)
  Using cached nvidia_cufft_cu12-11.0.2.54-py3-none-manylinux1_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-curand-cu12==10.3.2.106 (from torch>=1.11.0->ctgan>=0.10.0->sdv)
  Using cached nvidia_curand_cu12-10.3.2.106-py3-none-manylinux1_x86_64.whl.metadata (1.5 kB)
Collecting nvidia-cusolver-cu12==11.4.5.107 (from torch>=1.11.0->ctgan>=0.10.0->sdv)
  Using cached nvidia_cusolver_cu12-11.4.5.107-py3-none-manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-cusparse-cu12==12.1.0.106 (from torch>=1.11.0->ctgan>=0.10.0->sdv)
  Using cached nvidia_cusparse_cu12-12.1.0.106-py3-none-manylinux1_x86_64.whl.metadata (1.6 kB)
Collecting nvidia-nccl-cu12==2.20.5 (from torch>=1.11.0->ctgan>=0.10.0->sdv)
  Using cached nvidia_nccl_cu12-2.20.5-py3-none-manylinux2014_x86_64.whl.metadata (1.8 kB)
Collecting nvidia-nvtx-cu12==12.1.105 (from torch>=1.11.0->ctgan>=0.10.0->sdv)
  Using cached nvidia_nvtx_cu12-12.1.105-py3-none-manylinux1_x86_64.whl.metadata (1.7 kB)
Requirement already satisfied: triton==2.3.1 in /usr/local/lib/python3.10/dist-packages (from torch>=1.11.0->ctgan>=0.10.0->sdv) (2.3.1)
Collecting nvidia-nvjitlink-cu12 (from nvidia-cusolver-cu12==11.4.5.107->torch>=1.11.0->ctgan>=0.10.0->sdv)
  Downloading nvidia_nvjitlink_cu12-12.5.82-py3-none-manylinux2014_x86_64.whl.metadata (1.5 kB)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.11.0->ctgan>=0.10.0->sdv) (2.1.5)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.11.0->ctgan>=0.10.0->sdv) (1.3.0)
Downloading sdv-1.15.0-py3-none-any.whl (146 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 146.4/146.4 kB 10.2 MB/s eta 0:00:00
Downloading boto3-1.34.151-py3-none-any.whl (139 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 139.2/139.2 kB 13.2 MB/s eta 0:00:00
Downloading botocore-1.34.151-py3-none-any.whl (12.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.4/12.4 MB 115.6 MB/s eta 0:00:00
Downloading copulas-0.11.0-py3-none-any.whl (51 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 51.9/51.9 kB 4.6 MB/s eta 0:00:00
Downloading ctgan-0.10.1-py3-none-any.whl (24 kB)
Downloading deepecho-0.6.0-py3-none-any.whl (27 kB)
Downloading rdt-1.12.2-py3-none-any.whl (65 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 65.2/65.2 kB 6.4 MB/s eta 0:00:00
Downloading sdmetrics-0.15.0-py3-none-any.whl (170 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 170.5/170.5 kB 12.6 MB/s eta 0:00:00
Downloading Faker-26.0.0-py3-none-any.whl (1.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 78.2 MB/s eta 0:00:00
Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Downloading plotly-5.23.0-py3-none-any.whl (17.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 17.3/17.3 MB 38.2 MB/s eta 0:00:00
Downloading s3transfer-0.10.2-py3-none-any.whl (82 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 82.7/82.7 kB 8.0 MB/s eta 0:00:00
Using cached nvidia_cublas_cu12-12.1.3.1-py3-none-manylinux1_x86_64.whl (410.6 MB)
Using cached nvidia_cuda_cupti_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (14.1 MB)
Using cached nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (23.7 MB)
Using cached nvidia_cuda_runtime_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (823 kB)
Using cached nvidia_cudnn_cu12-8.9.2.26-py3-none-manylinux1_x86_64.whl (731.7 MB)
Using cached nvidia_cufft_cu12-11.0.2.54-py3-none-manylinux1_x86_64.whl (121.6 MB)
Using cached nvidia_curand_cu12-10.3.2.106-py3-none-manylinux1_x86_64.whl (56.5 MB)
Using cached nvidia_cusolver_cu12-11.4.5.107-py3-none-manylinux1_x86_64.whl (124.2 MB)
Using cached nvidia_cusparse_cu12-12.1.0.106-py3-none-manylinux1_x86_64.whl (196.0 MB)
Using cached nvidia_nccl_cu12-2.20.5-py3-none-manylinux2014_x86_64.whl (176.2 MB)
Using cached nvidia_nvtx_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (99 kB)
Downloading nvidia_nvjitlink_cu12-12.5.82-py3-none-manylinux2014_x86_64.whl (21.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.3/21.3 MB 10.5 MB/s eta 0:00:00
Installing collected packages: plotly, nvidia-nvtx-cu12, nvidia-nvjitlink-cu12, nvidia-nccl-cu12, nvidia-curand-cu12, nvidia-cufft-cu12, nvidia-cuda-runtime-cu12, nvidia-cuda-nvrtc-cu12, nvidia-cuda-cupti-cu
  Attempting uninstall: plotly
    Found existing installation: plotly 5.15.0
    Uninstalling plotly-5.15.0:
      Successfully uninstalled plotly-5.15.0
Successfully installed Faker-26.0.0 boto3-1.34.151 botocore-1.34.151 copulas-0.11.0 ctgan-0.10.1 deepecho-0.6.0 jmespath-1.0.1 nvidia-cublas-cu12-12.1.3.1 nvidia-cuda-cupti-cu12-12.1.105 nvidia-cuda-nvrtc-cu
```

```
pip install table_evaluator
```

```
Collecting table_evaluator
  Downloading table_evaluator-1.6.1-py3-none-any.whl.metadata (8.8 kB)
Collecting pandas==2.0.* (from table_evaluator)
  Downloading pandas-2.0.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (18 kB)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from table_evaluator) (1.26.4)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from table_evaluator) (4.66.4)
Requirement already satisfied: psutil in /usr/local/lib/python3.10/dist-packages (from table_evaluator) (5.9.5)
Collecting dython==0.7.3 (from table_evaluator)
  Downloading dython-0.7.3-py3-none-any.whl.metadata (3.0 kB)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (from table_evaluator) (0.13.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from table_evaluator) (3.7.1)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (from table_evaluator) (1.3.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from table_evaluator) (1.13.1)
Collecting scikit-plot>=0.3.7 (from dython==0.7.3->table_evaluator)
  Downloading scikit_plot-0.3.7-py3-none-any.whl.metadata (7.1 kB)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas==2.0.*->table_evaluator) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas==2.0.*->table_evaluator) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas==2.0.*->table_evaluator) (2024.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->table_evaluator) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->table_evaluator) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->table_evaluator) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->table_evaluator) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->table_evaluator) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->table_evaluator) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->table_evaluator) (3.1.2)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->table_evaluator) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->table_evaluator) (3.5.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas==2.0.*->table_evaluator) (1.16.0)
Downloading table_evaluator-1.6.1-py3-none-any.whl (22 kB)
Downloading dython-0.7.3-py3-none-any.whl (23 kB)
Downloading pandas-2.0.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.3/12.3 MB 106.0 MB/s eta 0:00:00
Downloading scikit_plot-0.3.7-py3-none-any.whl (33 kB)
Installing collected packages: pandas, scikit-plot, dython, table_evaluator
  Attempting uninstall: pandas
    Found existing installation: pandas 2.1.4
    Uninstalling pandas-2.1.4:
      Successfully uninstalled pandas-2.1.4
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts.
google-colab 1.0.0 requires pandas==2.1.4, but you have pandas 2.0.3 which is incompatible.
Successfully installed dython-0.7.3 pandas-2.0.3 scikit-plot-0.3.7 table_evaluator-1.6.1
```

```
import pandas as pd

from sdmetrics.reports.single_table import QualityReport
from ctgan import CTGAN

from rdt import HyperTransformer
```

```
real_data = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/CSV_FILE/adult.csv")
```

```
df = pd.DataFrame(real_data)

print(df.columns)

print("Original DataFrame:")
print(df)
```

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'education.num',
       'marital.status', 'occupation', 'relationship', 'race', 'sex',
       'capital.gain', 'capital.loss', 'hours.per.week', 'native.country',
       'income'],
      dtype='object')
Original DataFrame:
       age workclass  fnlwgt     education  education.num      marital.status  \
0       90         ?   77053       HS-grad              9             Widowed
1       82   Private  132870       HS-grad              9             Widowed
2       66         ?  186061  Some-college             10             Widowed
3       54   Private  140359       7th-8th              4            Divorced
4       41   Private  264663  Some-college             10           Separated
...    ...       ...     ...           ...            ...                 ...
32556   22   Private  310152  Some-college             10       Never-married
32557   27   Private  257302    Assoc-acdm             12  Married-civ-spouse
32558   40   Private  154374       HS-grad              9  Married-civ-spouse
32559   58   Private  151910       HS-grad              9             Widowed
32560   22   Private  201490       HS-grad              9       Never-married

              occupation   relationship   race     sex  capital.gain  \
0                      ?  Not-in-family  White  Female             0
1        Exec-managerial  Not-in-family  White  Female             0
2                      ?      Unmarried  Black  Female             0
3      Machine-op-inspct      Unmarried  White  Female             0
4         Prof-specialty      Own-child  White  Female             0
...                  ...            ...    ...     ...           ...
32556    Protective-serv  Not-in-family  White    Male             0
32557       Tech-support           Wife  White  Female             0
32558  Machine-op-inspct        Husband  White    Male             0
32559       Adm-clerical      Unmarried  White  Female             0
32560       Adm-clerical      Own-child  White    Male             0

       capital.loss  hours.per.week native.country income
0              4356              40  United-States  <=50K
1              4356              18  United-States  <=50K
2              4356              40  United-States  <=50K
3              3900              40  United-States  <=50K
4              3900              40  United-States  <=50K
...             ...             ...            ...    ...
32556             0              40  United-States  <=50K
32557             0              38  United-States  <=50K
32558             0              40  United-States   >50K
32559             0              40  United-States  <=50K
32560             0              20  United-States  <=50K

[32561 rows x 15 columns]
```

```
NUM_ROWS = 100000
NUM_EPOCHS = 1000
BATCH_SIZE = 1500
```

```
df.shape
```

```
(32561, 15)
```

```
ht = HyperTransformer()
ht.detect_initial_config(data = df)
detected_config = ht.get_config()
display(detected_config)
```

```
{
    "sdtypes": {
        "age": "numerical",
        "workclass": "categorical",
        "fnlwgt": "numerical",
        "education": "categorical",
        "education.num": "numerical",
        "marital.status": "categorical",
        "occupation": "categorical",
        "relationship": "categorical",
        "race": "categorical",
        "sex": "categorical",
        "capital.gain": "numerical",
        "capital.loss": "numerical",
        "hours.per.week": "numerical",
        "native.country": "categorical",
        "income": "categorical"
    },
    "transformers": {
        "age": FloatFormatter(),
        "workclass": UniformEncoder(),
        "fnlwgt": FloatFormatter(),
        "education": UniformEncoder(),
        "education.num": FloatFormatter(),
        "marital.status": UniformEncoder(),
        "occupation": UniformEncoder(),
        "relationship": UniformEncoder(),
        "race": UniformEncoder(),
        "sex": UniformEncoder(),
        "capital.gain": FloatFormatter(),
        "capital.loss": FloatFormatter(),
        "hours.per.week": FloatFormatter(),
        "native.country": UniformEncoder(),
        "income": UniformEncoder()
    }
}
```

```
ht.fit(df)
transformed_df = ht.transform(df)
transformed_df
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
0	90.0	0.052366	77053.0	0.053123	9.0	0.013082	0.019119	0.143596	0.373419	0.133523	0.0	4356.0	40.0	0.459319	0.272547
1	82.0	0.568465	132870.0	0.300369	9.0	0.017199	0.149767	0.245796	0.486551	0.192714	0.0	4356.0	18.0	0.708768	0.277429
2	66.0	0.022600	186061.0	0.504480	10.0	0.027261	0.014762	0.350066	0.925277	0.066922	0.0	4356.0	40.0	0.683127	0.708181
3	54.0	0.609363	140359.0	0.548441	4.0	0.105613	0.191911	0.287902	0.354498	0.319793	0.0	3900.0	40.0	0.603400	0.633314
4	41.0	0.470378	264663.0	0.415532	10.0	0.188802	0.283315	0.367358	0.644581	0.100573	0.0	3900.0	40.0	0.568972	0.108195
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
32556	22.0	0.578015	310152.0	0.458453	10.0	0.484006	0.989882	0.043013	0.749511	0.941382	0.0	0.0	40.0	0.467547	0.640913
32557	27.0	0.573395	257302.0	0.896722	12.0	0.787018	0.962568	0.980980	0.382848	0.320521	0.0	0.0	38.0	0.809435	0.750532
32558	40.0	0.659758	154374.0	0.040210	9.0	0.967204	0.196351	0.830186	0.000026	0.494150	0.0	0.0	40.0	0.172248	0.969286
32559	58.0	0.497835	151910.0	0.258776	9.0	0.027938	0.537457	0.284072	0.336407	0.269576	0.0	0.0	40.0	0.761467	0.464751
32560	22.0	0.697324	201490.0	0.249631	9.0	0.433607	0.553605	0.439984	0.420466	0.496134	0.0	0.0	20.0	0.105233	0.692251

32561 rows × 15 columns

Next steps: Generate code with `transformed_df` | View recommended plots | New interactive sheet

```
import time

start_time = time.time()  # Capture start time before training

model = CTGAN(
    epochs=NUM_EPOCHS,
    verbose=True,
    batch_size=BATCH_SIZE,
    embedding_dim = 1024,
    discriminator_steps = 6,
    discriminator_dim = (512,512)
)

model.fit(transformed_df)

# Training is finished, record end time
end_time = time.time()

# Calculate total training time in seconds
training_time = end_time - start_time

print(f"Training completed! Total time taken: {training_time:.2f} seconds")

model.save("/content/drive/MyDrive/Colab Notebooks/CSV_FILE/Models/adult_1000epochs_1500BS_1024_6_512.pkl")
```

```
Gen. (0.00) | Discrim. (0.00):   0%|          | 0/1000 [00:00<?, ?it/s]/usr/local/lib/python3.10/dist-packages/torch/autograd/graph.py:744: UserWarning: Attempting to run cuBLAS, but there was no current CUDA c
  return Variable._execution_engine.run_backward(  # Calls into the C++ engine to run the backward pass
Gen. (-1.51) | Discrim. (-0.33): 100%|██████████| 1000/1000 [38:38<00:00,  2.32s/it]
Training completed! Total time taken: 2377.96 seconds
```

```
from sdv.metadata import SingleTableMetadata
metadata = SingleTableMetadata()
metadata.detect_from_dataframe(df)
metadata_dict= metadata.to_dict()
metadata.visualize()
```

age : numerical
workclass : categorical
fnlwgt : numerical
education : categorical
education.num : numerical
marital.status : categorical
occupation : categorical
relationship : categorical
race : categorical
sex : categorical
capital.gain : numerical
capital.loss : numerical
hours.per.week : numerical
native.country : categorical
income : categorical

```
categorical_columns = [column for column, info in metadata_dict['columns'].items() if info['sdtype'] == 'categorical']
print(categorical_columns)
```

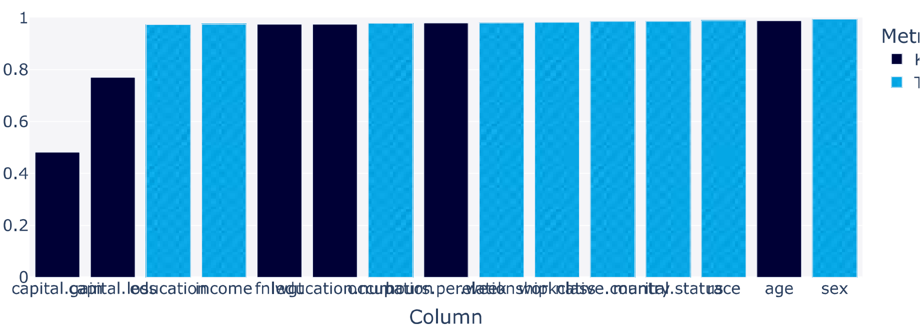
```
['workclass', 'education', 'marital.status', 'occupation', 'relationship', 'race', 'sex', 'native.country', 'income']
```

```
from sdmetrics.reports.single_table import QualityReport

# Get Synthetic data
synthetic_data = model.sample(NUM_ROWS)
# reverse transform the data
synthetic_data = ht.reverse_transform(synthetic_data)

report = QualityReport()
# Use the metadata OBJECT instead of the dictionary
report.generate(df, synthetic_data, metadata.to_dict())

cs_report = report.get_details(property_name="Column Shapes")
print(cs_report)

# Create the first figure
fig1 = report.get_visualization(property_name='Column Shapes')
fig1.show()

# Create the second figure
fig2 = report.get_visualization(property_name='Column Pair Trends')

fig2.show()

report.save(filepath='/content/drive/MyDrive/Colab Notebooks/CSV_FILE/Models/adult_report_1000epochs_1500BS_1024_6_512.pkl')
```

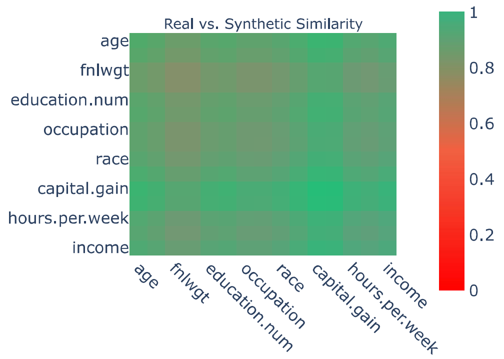
```
Generating report ...

(1/2) Evaluating Column Shapes: |██████████| 15/15 [00:00<00:00, 37.17it/s]|
Column Shapes Score: 93.41%

(2/2) Evaluating Column Pair Trends: |██████████| 105/105 [00:07<00:00, 14.25it/s]|
Column Pair Trends Score: 89.63%

Overall Score (Average): 91.52%
```

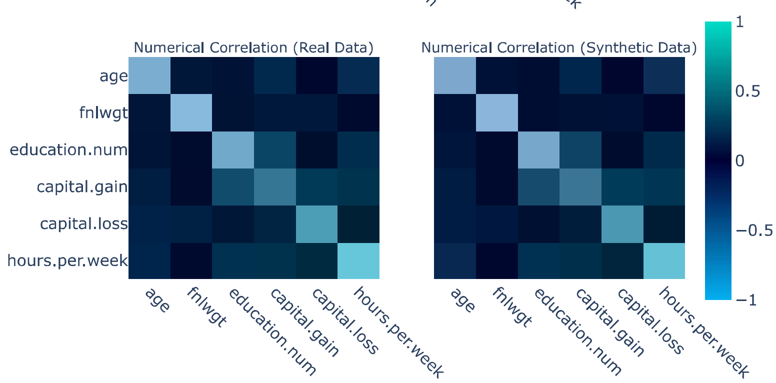
	Column	Metric	Score
0	age	KSComplement	0.988581
1	workclass	TVComplement	0.982307
2	fnlwgt	KSComplement	0.974960
3	education	TVComplement	0.973206
4	education.num	KSComplement	0.975200
5	marital.status	TVComplement	0.985605
6	occupation	TVComplement	0.977816
7	relationship	TVComplement	0.980158
8	race	TVComplement	0.988574
9	sex	TVComplement	0.993995
10	capital.gain	KSComplement	0.481810
11	capital.loss	KSComplement	0.770340
12	hours.per.week	KSComplement	0.979356
13	native.country	TVComplement	0.985305
14	income	TVComplement	0.974960

Data Quality: Column Shapes (Average Score=0.93)



Data Quality: Column Pair Trends (Average Score=0.9)





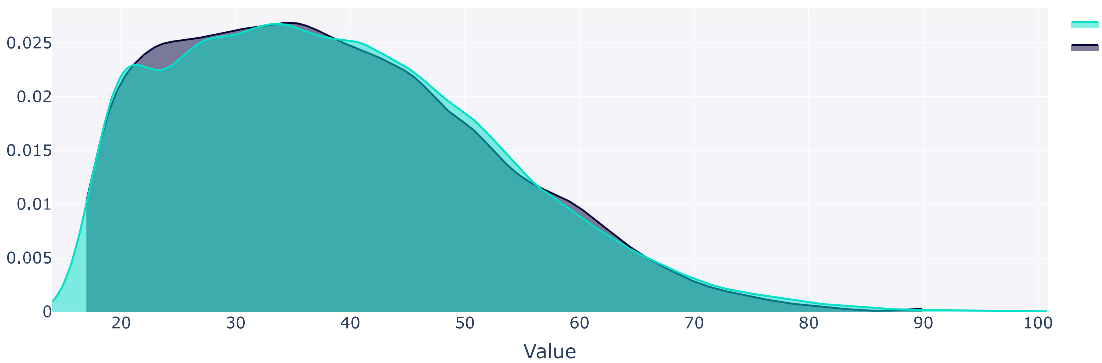

```python
from sdmetrics.single_column import CSTest

for column in categorical_columns:
    cstest_result = CSTest.compute(
        real_data=df[column],
        synthetic_data=synthetic_data[column]
    )
    print(f"CSTest for column {column}: {cstest_result}")
```

```
CSTest for column workclass: 0.9997086201516404
CSTest for column education: 1.0
CSTest for column marital.status: 0.9999572209453049
CSTest for column occupation: 1.0
CSTest for column relationship: 0.9999999505238084
CSTest for column race: 0.9999977261147913
CSTest for column sex: 0.9898160112148474
```

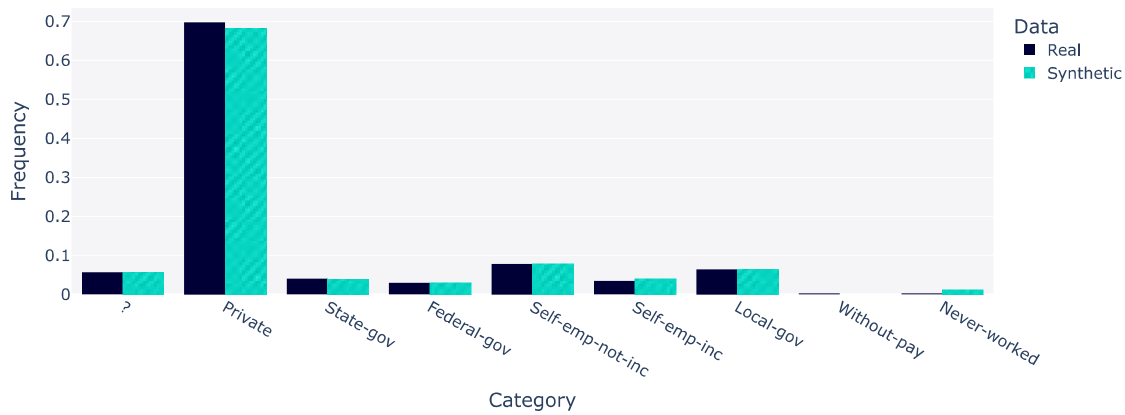
```
CSTest for column native.country: 1.0
CSTest for column income: 0.9532994881448515
```

```
from sdmetrics.visualization import get_column_plot

# Loop through each column in the dataframe
for column in df.columns:
    fig = get_column_plot(
        real_data=df,
        synthetic_data=synthetic_data,
        column_name=column,
    )

    fig.show()
```

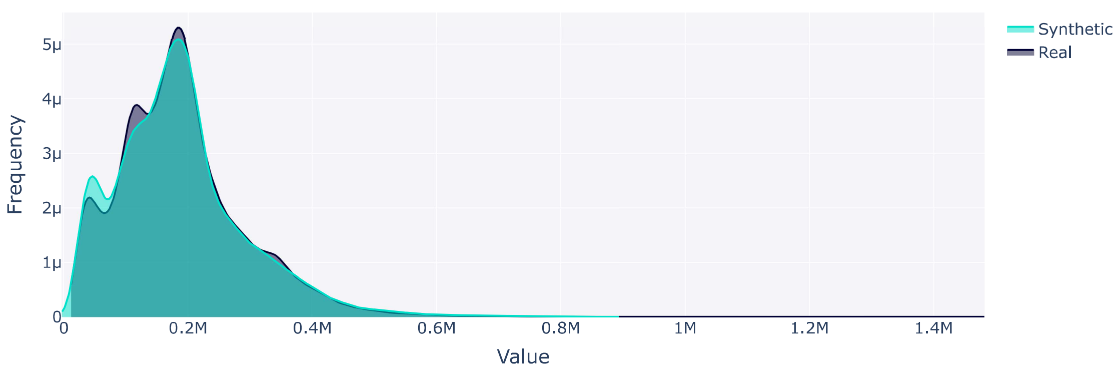
Real vs. Synthetic Data for column 'age'



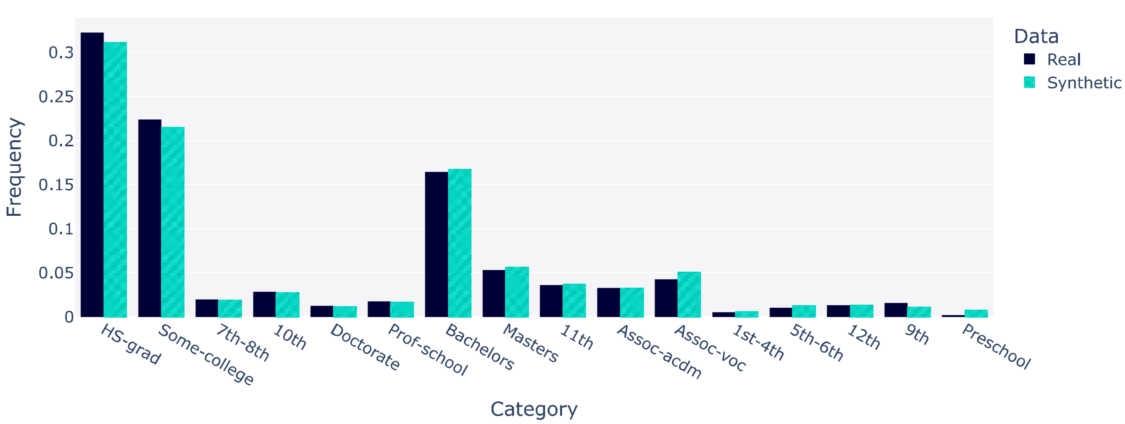
Real vs. Synthetic Data for column 'workclass'



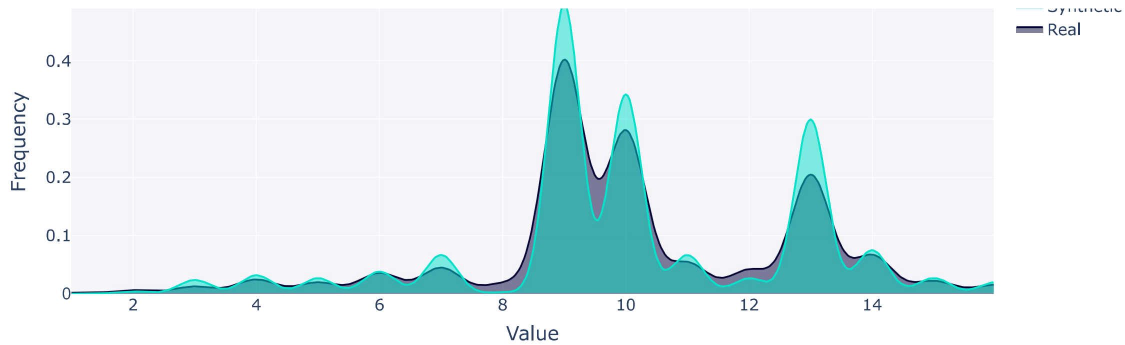
Real vs. Synthetic Data for column 'fnlwgt'



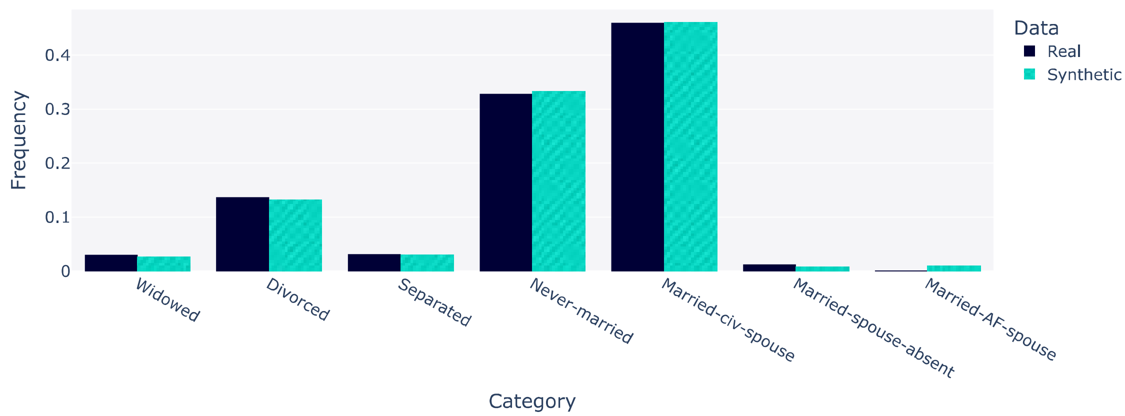
Real vs. Synthetic Data for column 'education'



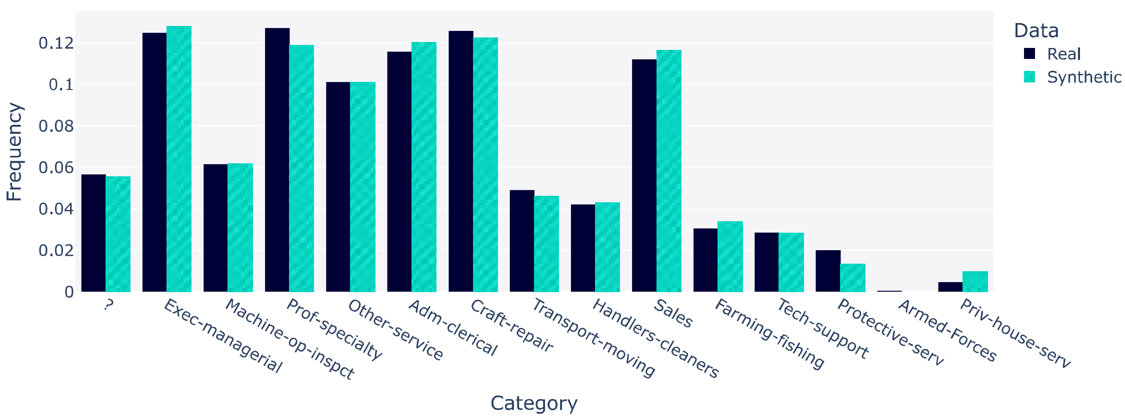
Real vs. Synthetic Data for column 'education.num'



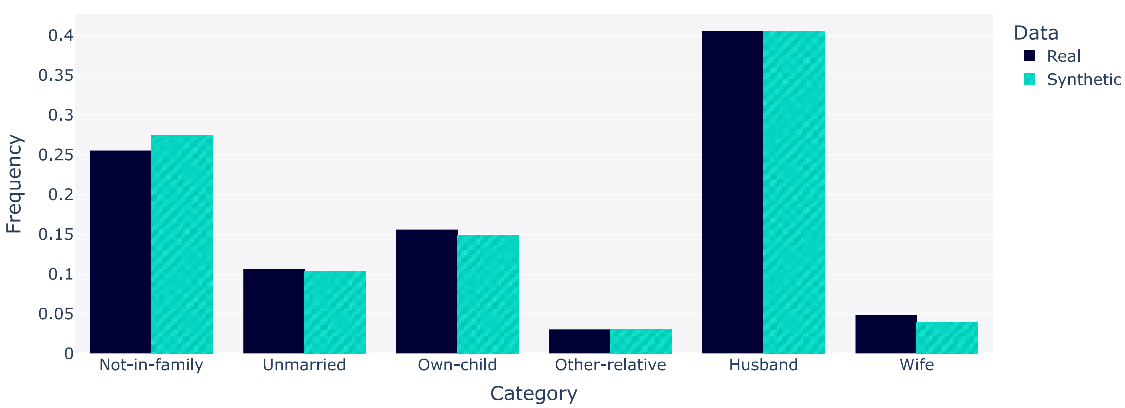
Real vs. Synthetic Data for column 'marital.status'



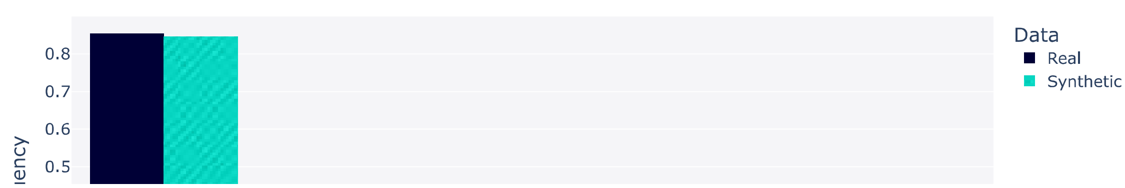
Real vs. Synthetic Data for column 'occupation'

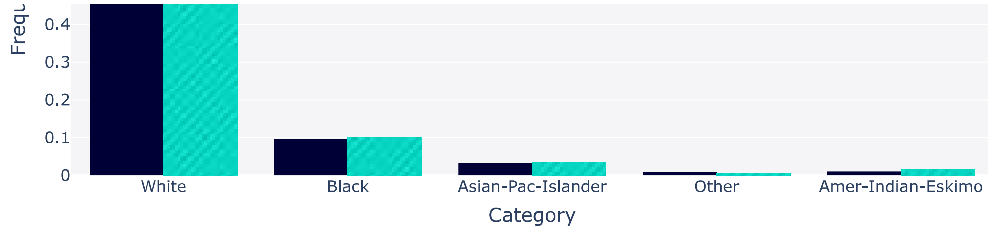


Real vs. Synthetic Data for column 'relationship'

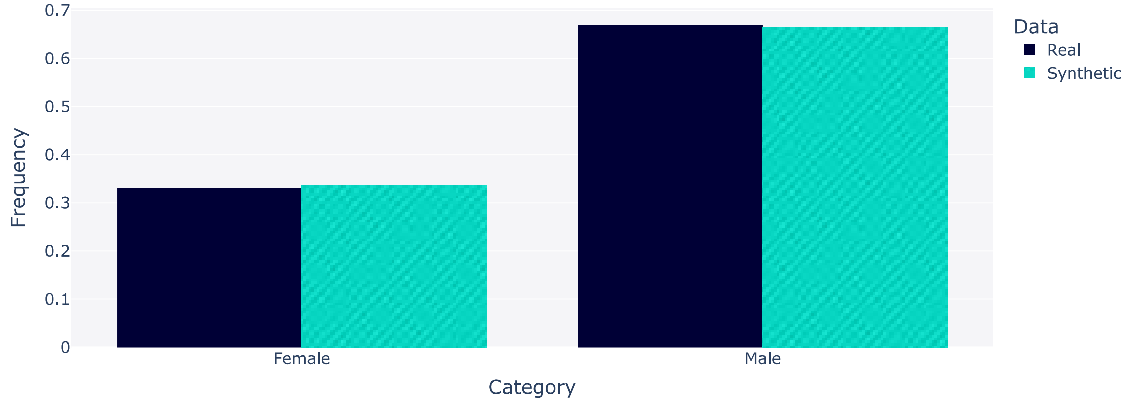


Real vs. Synthetic Data for column 'race'

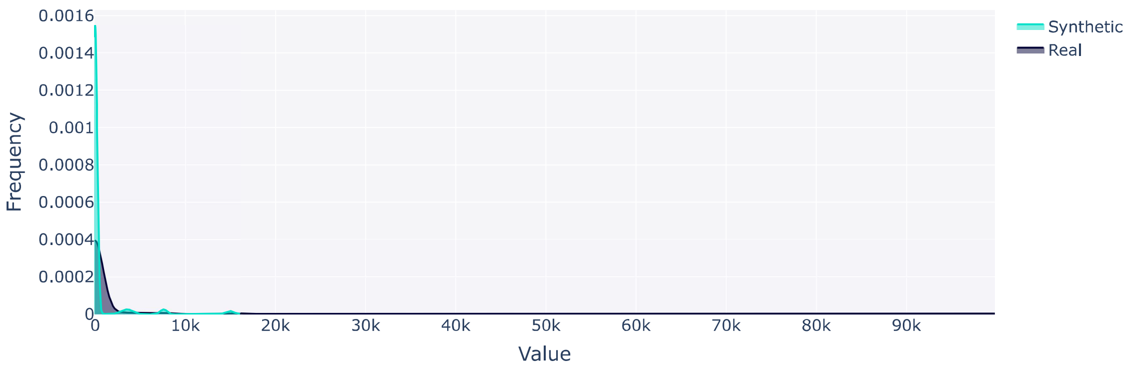




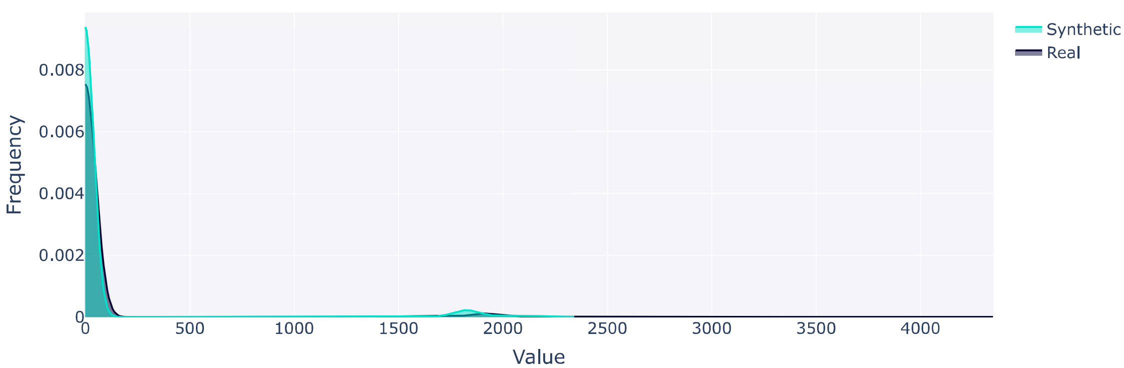
Real vs. Synthetic Data for column 'sex'



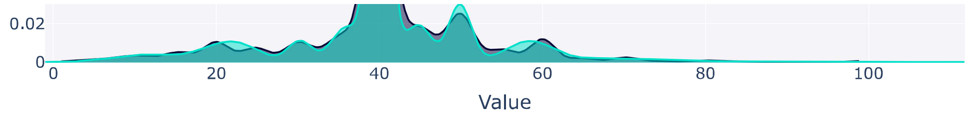
Real vs. Synthetic Data for column 'capital.gain'



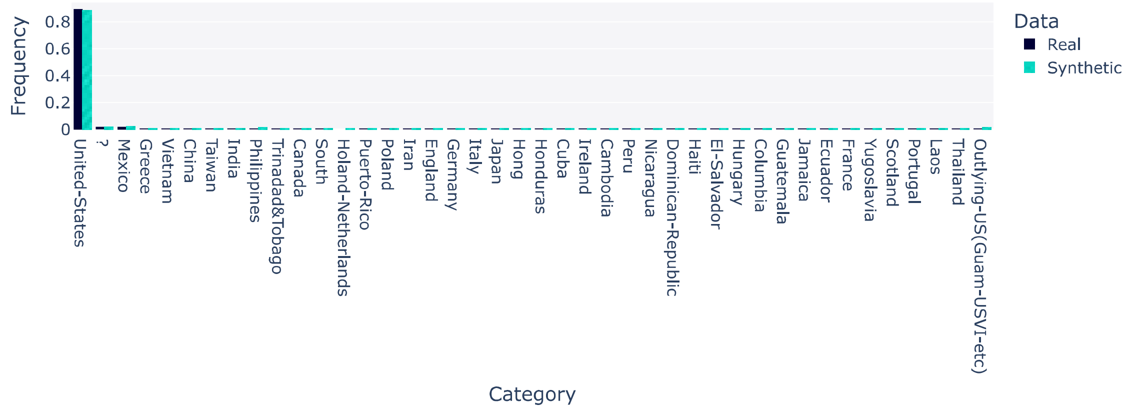
Real vs. Synthetic Data for column 'capital.loss'



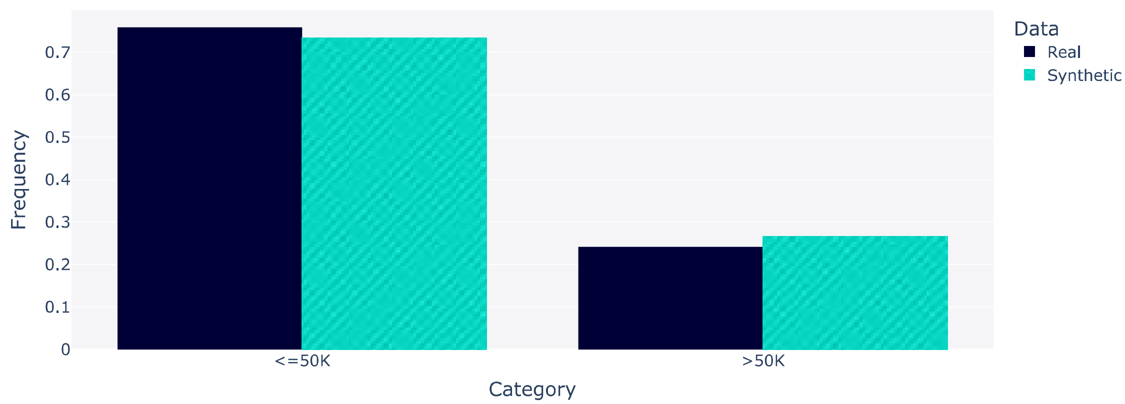
Real vs. Synthetic Data for column 'hours.per.week'



Real vs. Synthetic Data for column 'native.country'



Real vs. Synthetic Data for column 'income'



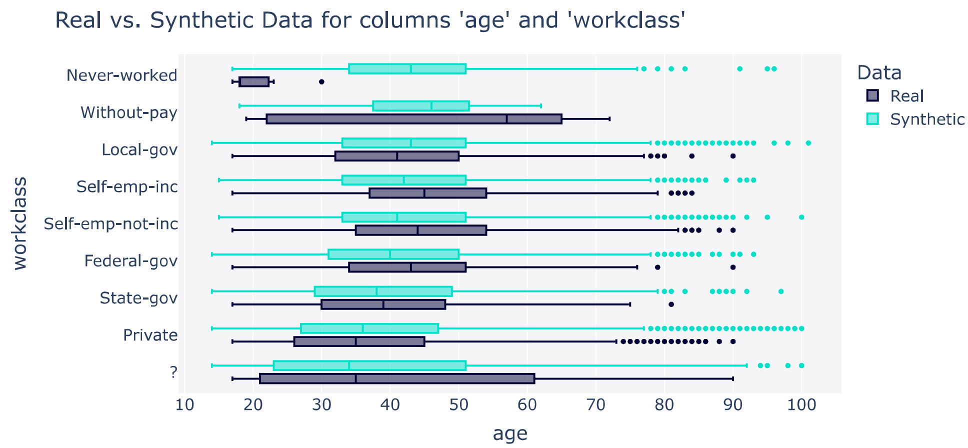

```
display(synthetic_data)
```

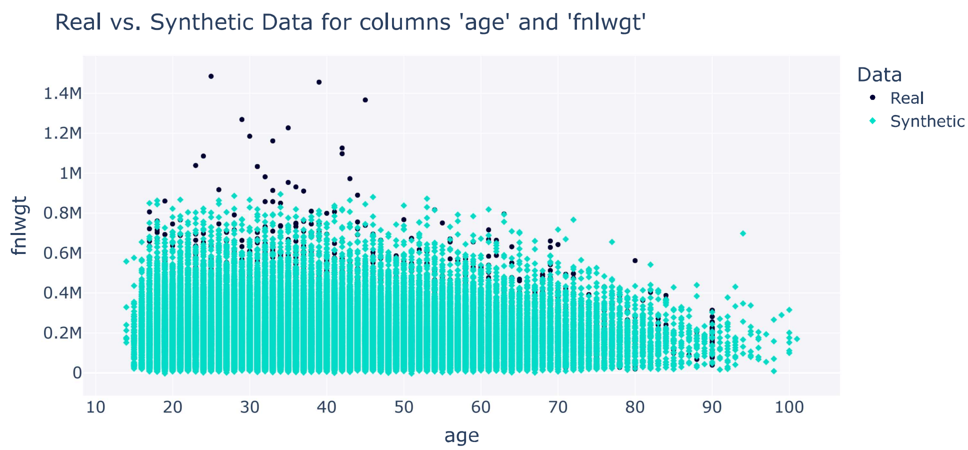
	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
0	25	Private	182573	5th-6th	11	Never-married	Adm-clerical	Own-child	White	Female	-32	1	24	United-States	<=50K
1	68	Private	402324	Preschool	3	Married-civ-spouse	Exec-managerial	Husband	White	Male	-1	1	40	United-States	>50K
2	39	Private	121603	Some-college	10	Married-civ-spouse	Adm-clerical	Unmarried	White	Female	-7	0	40	Mexico	>50K
3	61	Federal-gov	118559	HS-grad	9	Married-civ-spouse	Adm-clerical	Husband	White	Male	-19	-1	40	?	<=50K
4	43	Private	277336	Preschool	11	Married-civ-spouse	Handlers-cleaners	Husband	White	Male	20	0	40	United-States	>50K
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
99995	34	Self-emp-not-inc	221206	HS-grad	9	Married-civ-spouse	?	Own-child	White	Male	2659	-1	15	United-States	<=50K
99996	31	Private	100825	9th	11	Separated	Exec-managerial	Not-in-family	White	Male	-19	0	40	United-States	<=50K
99997	33	Private	176971	HS-grad	9	Separated	Sales	Not-in-family	White	Female	31	0	40	United-States	<=50K
99998	38	Private	370078	Some-college	10	Married-civ-spouse	Exec-managerial	Wife	White	Male	-14	0	40	El-Salvador	<=50K
99999	27	Private	185228	Bachelors	13	Divorced	Craft-repair	Not-in-family	White	Female	-22	-1	40	United-States	<=50K

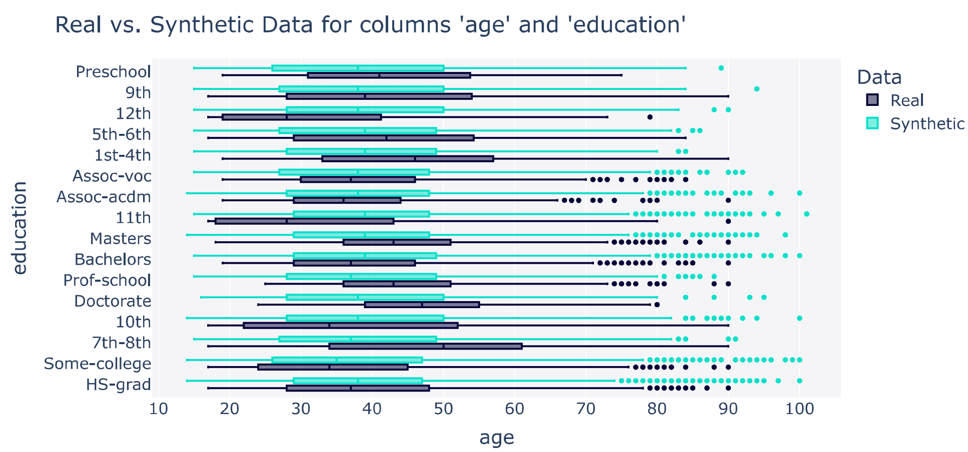
100000 rows × 15 columns

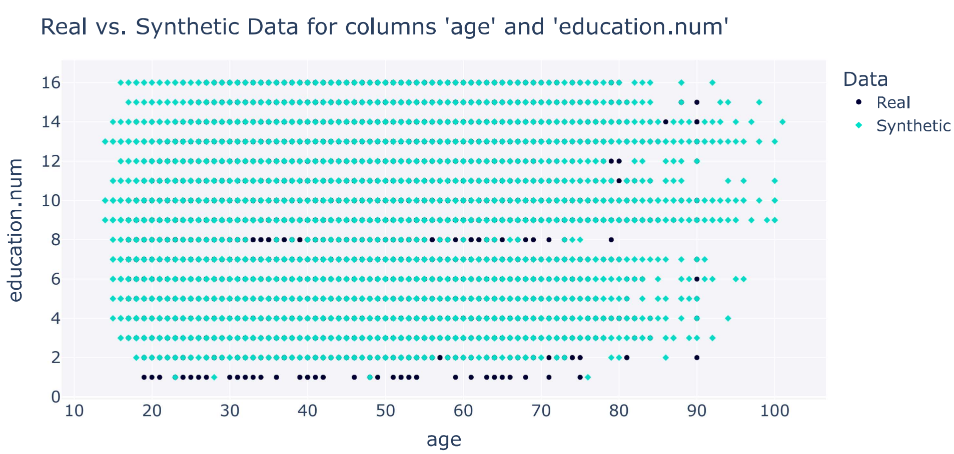
Next steps: Generate code with synthetic_data | View recommended plots | New interactive sheet

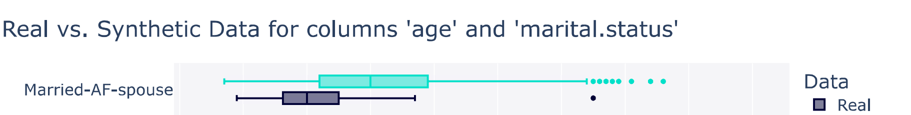
```
from itertools import combinations
from sdmetrics.visualization import get_column_pair_plot

# Get all column pairs
column_pairs = combinations(df.columns, 2)

# Loop through each column pair
for column1, column2 in column_pairs:
  # Generate the plot using get_column_pair_plot
  fig = get_column_pair_plot(
      real_data=df,
      synthetic_data=synthetic_data,
      column_names=[column1, column2]
  )


  fig.show()
```

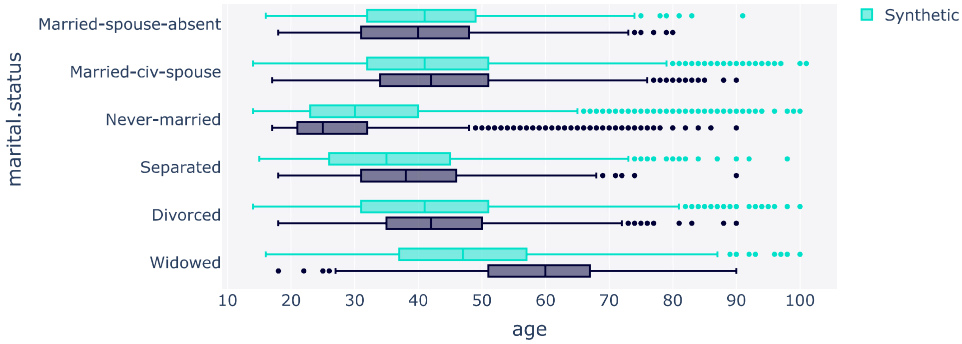

Real vs. Synthetic Data for columns 'age' and 'workclass'
workclass
Never-worked
Without-pay
Local-gov
Self-emp-inc
Self-emp-not-inc
Federal-gov
State-gov
Private
?
10 20 30 40 50 60 70 80 90 100
age
Data
Real
Synthetic


Real vs. Synthetic Data for columns 'age' and 'fnlwgt'
fnlwgt
1.4M
1.2M
1M
0.8M
0.6M
0.4M
0.2M
0
10 20 30 40 50 60 70 80 90 100
age
Data
Real
Synthetic

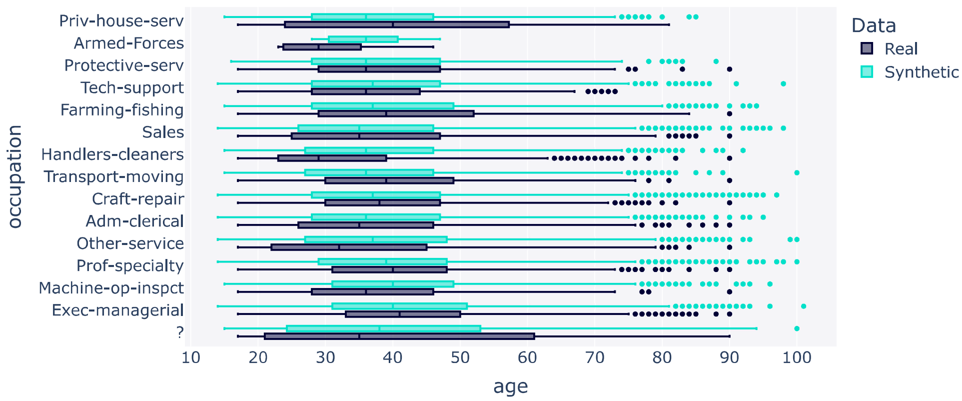

Real vs. Synthetic Data for columns 'age' and 'education'
education
Preschool
9th
12th
5th-6th
1st-4th
Assoc-voc
Assoc-acdm
11th
Masters
Bachelors
Prof-school
Doctorate
10th
7th-8th
Some-college
HS-grad
10 20 30 40 50 60 70 80 90 100
age
Data
Real
Synthetic


Real vs. Synthetic Data for columns 'age' and 'education.num'
education.num
16
14
12
10
8
6
4
2
0
10 20 30 40 50 60 70 80 90 100
age
Data
Real
Synthetic

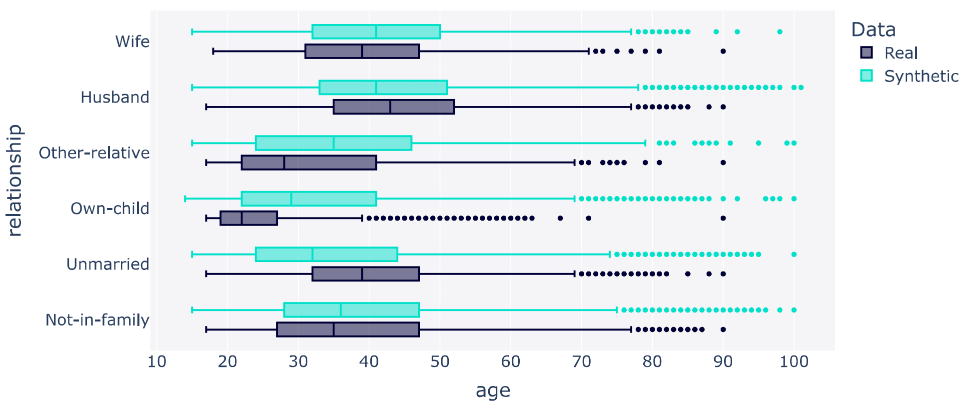

Real vs. Synthetic Data for columns 'age' and 'marital.status'
Married-AF-spouse
Data
Real



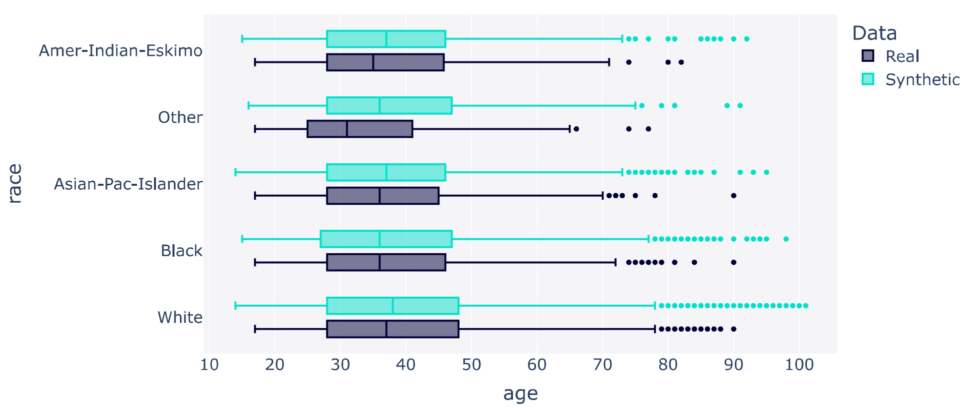
Real vs. Synthetic Data for columns 'age' and 'occupation'



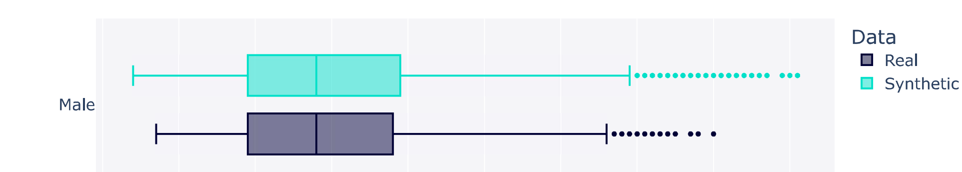
Real vs. Synthetic Data for columns 'age' and 'relationship'

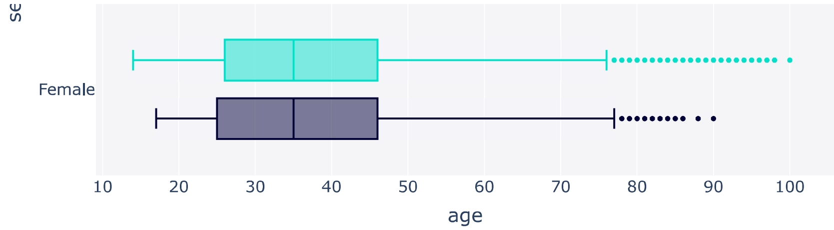


Real vs. Synthetic Data for columns 'age' and 'race'

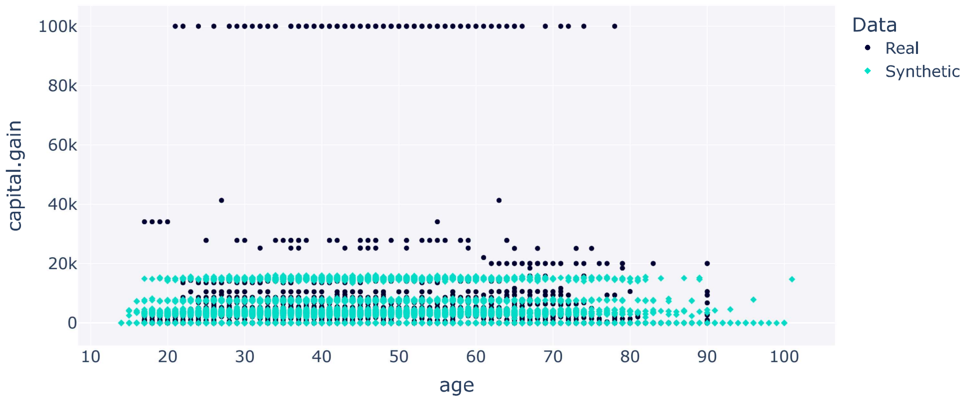


Real vs. Synthetic Data for columns 'age' and 'sex'

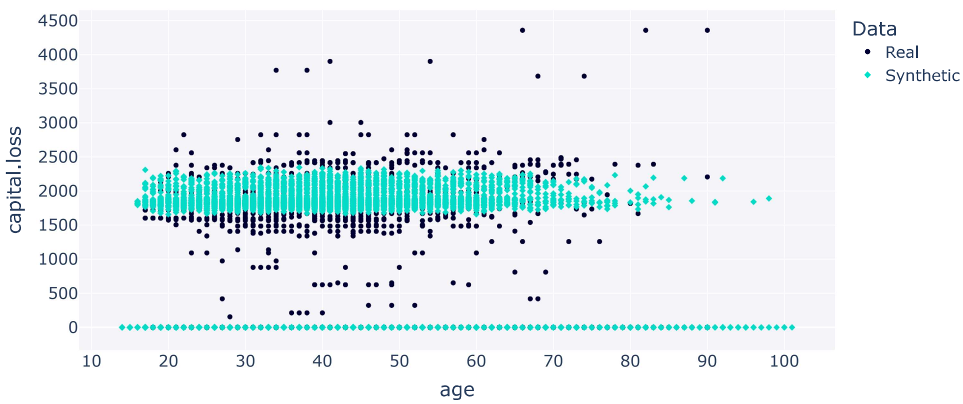




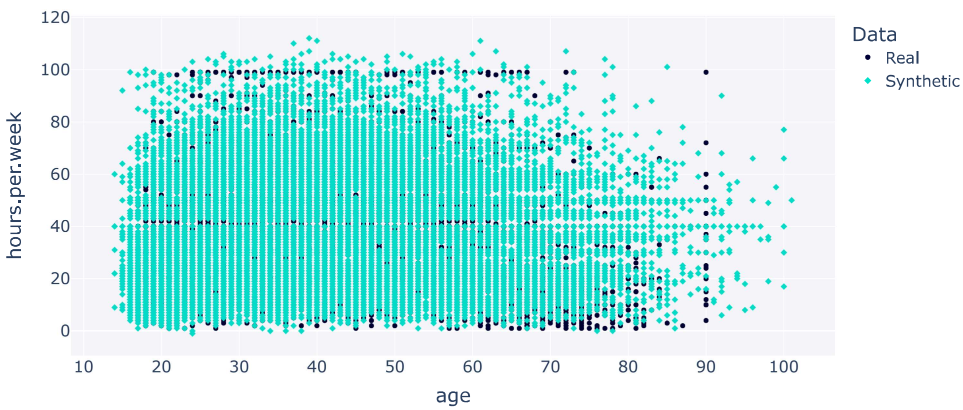
Real vs. Synthetic Data for columns 'age' and 'capital.gain'



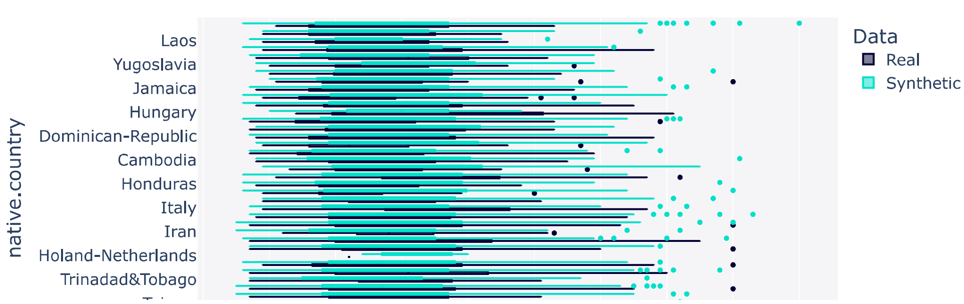
Real vs. Synthetic Data for columns 'age' and 'capital.loss'

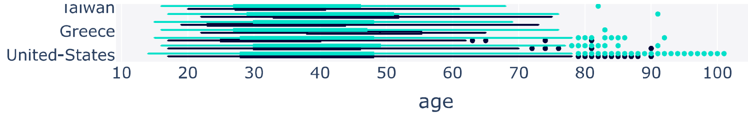


Real vs. Synthetic Data for columns 'age' and 'hours.per.week'

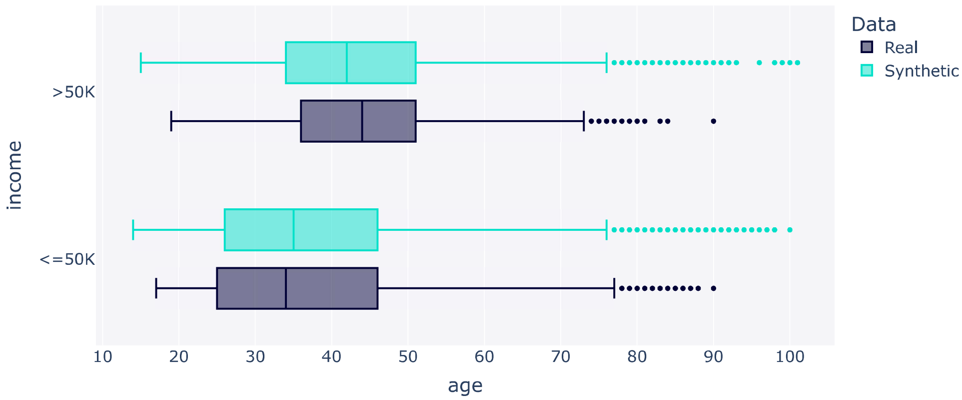


Real vs. Synthetic Data for columns 'age' and 'native.country'

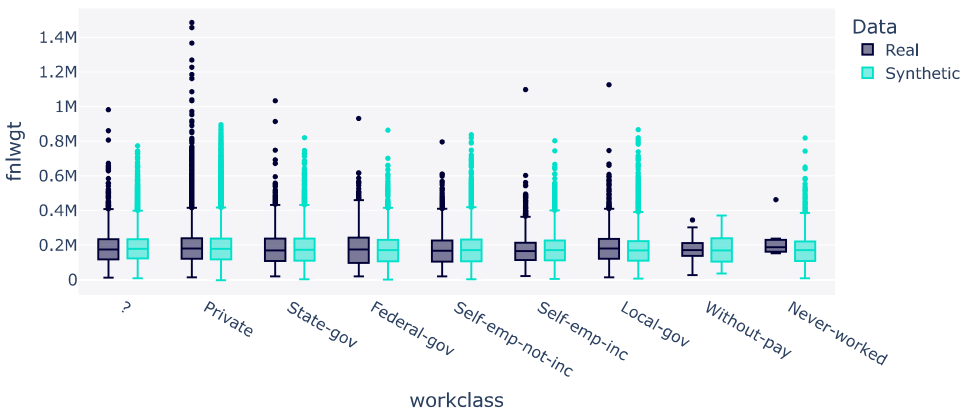




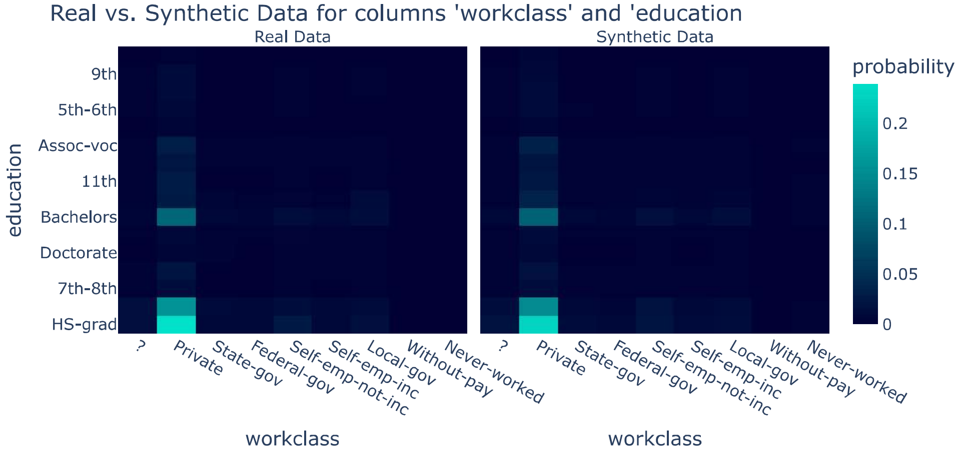
Real vs. Synthetic Data for columns 'age' and 'income'



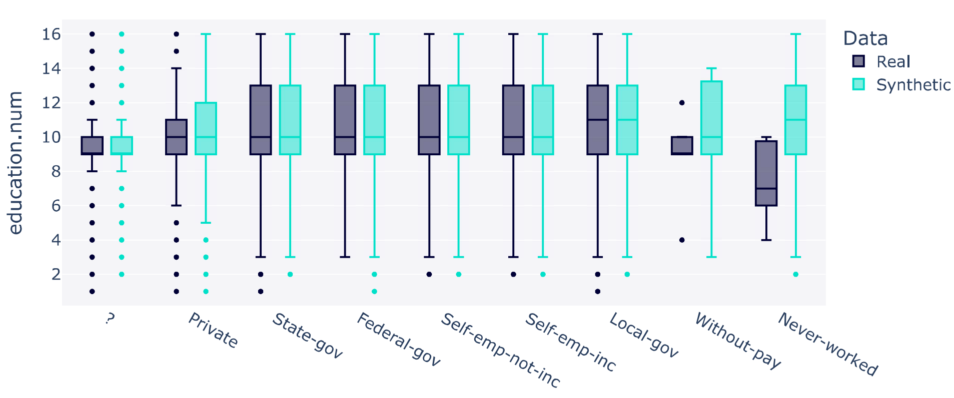
Real vs. Synthetic Data for columns 'workclass' and 'fnlwgt'

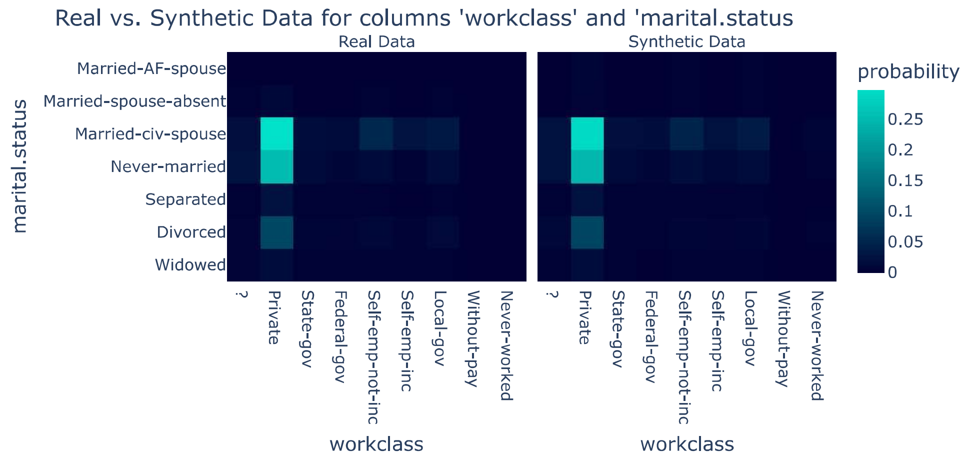


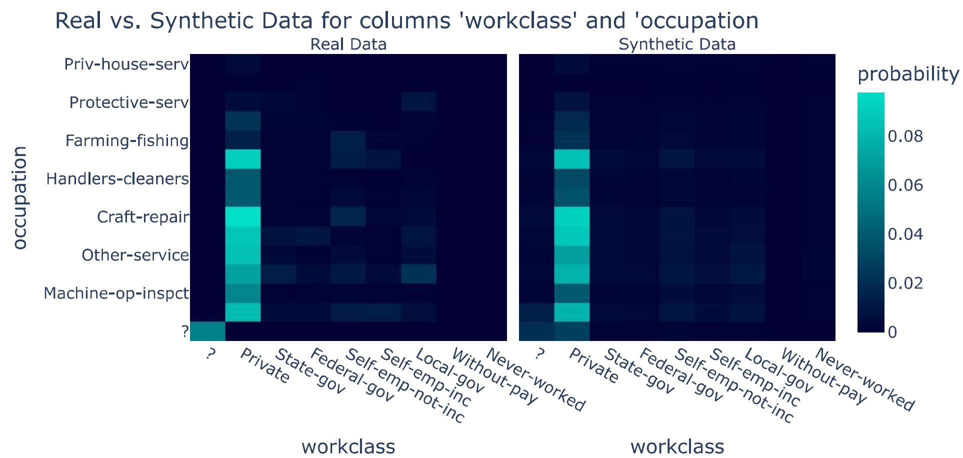
Real vs. Synthetic Data for columns 'workclass' and 'education

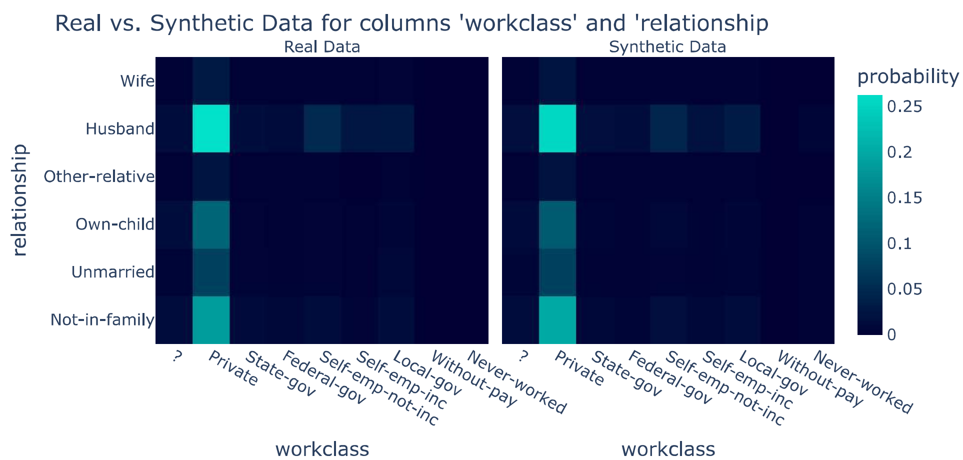


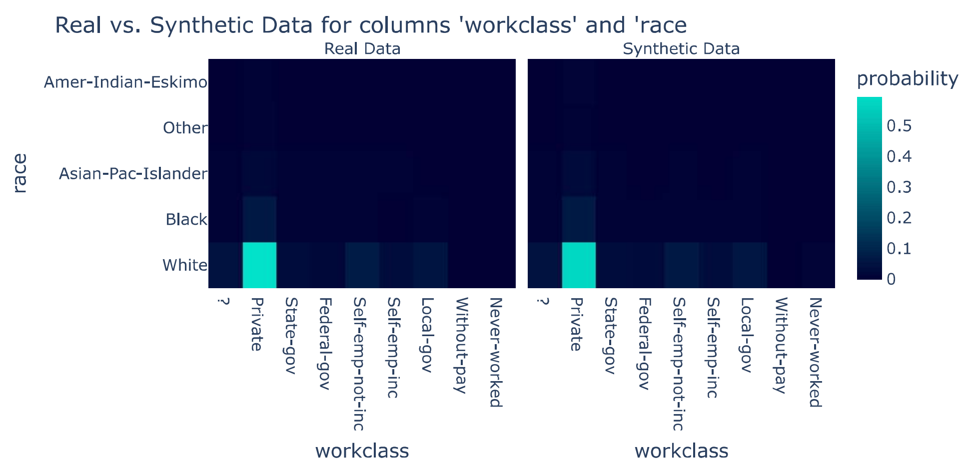
Real vs. Synthetic Data for columns 'workclass' and 'education.num'











Buffered data was truncated after reaching the output size limit.

```
from table_evaluator import TableEvaluator

# Assuming real_data and synthetic_data are pandas DataFrames
table_evaluator = TableEvaluator(df, synthetic_data)
```

```
table_evaluator.visual_evaluation()
```

...