

IMPLEMENTASI PIPELINE GENERASI DATA SINTETIS MENGGUNAKAN CTGAN

Dosen Pengampu Mata Kuliah: Ir. Sidik Prabowo, S.T., M.T., CEH, IDPP



Disusun Oleh:

Muhammad Karov Ardava Barus	103052300001
Muhammad Al Fayyedh Denof	103052330042
Avatar Bintang Ramadhan	103052300007
Runa Raditya Rizki Hidayat	103052300037

PROGRAM STUDI S1 SAINS DATA
FAKULTAS INFORMATIKA
UNIVERSITAS TELKOM
BANDUNG
2025

Daftar Isi

I. Pendahuluan	3
1.1 Latar Belakang	3
1.2 Tujuan	3
II. Implementasi Teknis	3
2.1 Preprocessing Data	3
2.2 Pelatihan Model (Training)	3
2.3 Generasi Data (Sampling)	4
III. Hasil Implementasi	4
3.1 Proses Training	4
3.2 Sampel Data Sintetis	4
3.3 Output File	5
IV. Kesimpulan	5

I. Pendahuluan

1.1 Latar Belakang

Pada minggu sebelumnya, kami telah merancang alur kerja (*pipeline*) untuk pembuatan data sintetis guna mengatasi masalah privasi dan ketidakseimbangan kelas pada dataset *Telco Customer Churn*. Minggu ke-4 ini berfokus pada tahap eksekusi teknis, yaitu mengimplementasikan rancangan tersebut ke dalam kode program menggunakan bahasa Python dan pustaka *Synthetic Data Vault* (SDV).

1.2 Tujuan

Tujuan dari laporan progres minggu ke-4 ini adalah:

1. Melakukan instalasi dan konfigurasi lingkungan pengembangan.
2. Melakukan *preprocessing* data untuk persiapan pelatihan model.
3. Melatih model *Conditional Tabular GAN* (CTGAN) menggunakan data asli.
4. Membangkitkan (*generate*) dataset sintetis sebanyak 2.000 sampel.

II. Implementasi Teknis

2.1 Preprocessing Data

Sebelum data dimasukkan ke dalam model, dilakukan beberapa tahapan pembersihan:

1. Penghapusan Identitas Langsung: Kolom *customerID* dihapus karena merupakan *Direct Identifier* yang unik untuk setiap pengguna dan tidak boleh dipelajari oleh model generatif.
2. Konversi Tipe Data: Kolom *TotalCharges* dipastikan bertipe numerik, dan nilai kosong (*missing values*) diisi dengan nilai rata-rata (*mean imputation*).

```
1 # Hapus Direct Identifiers
2 df_train = df.drop(columns=['customerID'])
3
4 # Handling Missing Values
5 df_train['TotalCharges'] = pd.to_numeric(df_train['TotalCharges'],
6 errors='coerce')
7 df_train['TotalCharges'].fillna(df_train['TotalCharges'].mean(),
8 inplace=True)
```

python

2.2 Pelatihan Model (Training)

Kami menggunakan algoritma *CTGANSynthesizer*. Metadata tabel (tipe data setiap kolom) dideteksi secara otomatis oleh library *sdv*. Model dilatih selama 100 epoch untuk memastikan model mempelajari distribusi data dengan cukup baik.

```
1 from sdv.single_table import CTGANSynthesizer
2 from sdv.metadata import SingleTableMetadata
3
4 # Deteksi Metadata
5 metadata = SingleTableMetadata()
```

python

```

6  metadata.detect_from_dataframe(df_train)
7
8  # Inisialisasi dan Training
9  synthesizer = CTGANSynthesizer(metadata, epochs=100, verbose=True)
10 synthesizer.fit(df_train)

```

2.3 Generasi Data (Sampling)

Setelah model dilatih, kami membangkitkan data sintetis. Sesuai dengan persyaratan tugas (minimal 1.500 sampel), kami membangkitkan 2.000 baris data.

```

1  # Generate 2000 baris data
2  n_samples = 2000
3  synthetic_data = synthesizer.sample(num_rows=n_samples)
4
5  # Simpan ke CSV
6  synthetic_data.to_csv('output/synthetic_telco_churn.csv', index=False)

```

python

III. Hasil Implementasi

3.1 Proses Training

Proses pelatihan berjalan lancar dengan indikator *loss* untuk Generator dan Discriminator yang terpantau stabil selama 100 epoch.

```

1  --- 2. Training CTGAN Model ---
2  Metadata terdeteksi:
3  {
4      'METADATA_SPEC_VERSION': 'SINGLE_TABLE_V1',
5      'columns': {
6          'gender': {'sdtype': 'categorical'},
7          'SeniorCitizen': {'sdtype': 'categorical'},
8          ...
9          'Churn': {'sdtype': 'categorical'}
10     }
11 }
12
13 Gen. (-1.69) | Discrim. (-0.02): 100%|██████████| 100/100 [01:22<00:00,
14 1.22it/s]
14 Pelatihan model selesai.

```

Text

3.2 Sampel Data Sintetis

Berikut adalah cuplikan 5 baris pertama dari dataset sintetis yang berhasil dibangkitkan.

Gen	Snr	Ptn	Dep	Tnr	Phn	Mul	Int	Sec	Bkp	Dev	Tch	TV	Mov	Ctr	Ppr	Pay	Mth	Tot	Chn
Male	1	Yes	No	72	Yes	No	DSL	No	Yes	No	No	No	No	Mth	Yes	Mail	24.7	776.8	No
Fem	0	Yes	No	72	Yes	Yes	Fbr	Yes	Yes	Yes	Yes	Yes	Yes	2Yr	Yes	Bank	101.6	8684	No
Male	1	Yes	No	39	Yes	No	Fbr	Yes	Yes	Yes	Yes	No	Yes	1Yr	No	Elec	64.0	2141	No
Male	0	Yes	Yes	8	Yes	No	No	No	No	No	No	No	No	1Yr	No	Mail	28.7	18.8	No
Fem	0	No	No	6	Yes	No	DSL	No	No	No	No	No	No	Mth	Yes	Elec	39.5	1792	Yes

Table 1: Preview Dataset Sintetis Hasil Generasi (Data disingkat agar muat dalam tabel)

Keterangan Singkatan: Gen: Gender, Snr: SeniorCitizen, Ptn: Partner, Dep: Dependents, Tnr: Tenure, Phn: PhoneService, Mul: MultipleLines, Int: InternetService, Sec: OnlineSecurity, Bkp: OnlineBackup, Dev: DeviceProtection, Tch: TechSupport, TV: StreamingTV, Mov: StreamingMovies, Ctr: Contract, Ppr: PaperlessBilling, Pay: PaymentMethod, Mth: MonthlyCharges, Tot: TotalCharges, Chn: Churn.

3.3 Output File

File hasil generasi telah disimpan dengan nama `synthetic_telco_churn.csv` di dalam folder `output/`. File ini siap digunakan untuk tahap evaluasi utilitas dan privasi pada minggu berikutnya.

IV. Kesimpulan

Implementasi *pipeline* generasi data sintetis pada Minggu 4 telah berhasil dilakukan. Model CTGAN sukses dilatih menggunakan dataset *Telco Customer Churn* yang telah diproses, dan menghasilkan 2.000 baris data sintetis. Langkah selanjutnya adalah melakukan evaluasi mendalam untuk membandingkan kualitas statistik dan privasi antara data asli dan data sintetis.