Aplikasi Generasi Data Sintesis

Tessa Kania Sagala[1], Renisha Putri Giani[2], Rian Bintang Wijaya[3], Gymnastiar Al Khoarizmy[4], Virdio Samuel Saragih[5]

Jurusan Sains Data, Fakultas Sains, Institut Teknologi Sumatera, Lampung Selatan, Indonesia

Email: tessa.122450040@student.itera.ac.id, renisha.122450079@student.itera.ac.id, rian.122450094@student.itera.ac.id, gymnastiar.122450096@student.itera.ac.id, virdio.122450124@student.itera.ac.id

1. Pendahuluan

Pada zaman digital yang semakin pesat ini, data menjadi sangat penting dari berbagai aspek, seperti misalnya pada pemerintahan, bisnis, kesehatan dan yang lainnya. Dari model kecerdasan buatan sampai dengan pengambilan keputusan yang berlandaskan pada data, ketersediaan data yang berkualitas menjadi sangat penting saat ini. Tapi untuk bisa mendapatkan akses ke data yang memadai tersebut tidaklah mudah, terutama untuk data yang sensitif atau dapat dikatakan langka. Maka dari itu generasi data sensitif merupakan solusi untuk permasalahan ini.

Aplikasi Generasi Data Sintesis merupakan sebuah teknik yang digunakan untuk membuat data baru yang menyerupai data asli dan berfungsi sebagai alternatif dari kumpulan data tersebut. Teknik ini digunakan untuk menciptakan dataset yang memiliki struktur, pola dan distribusi yang mirip dengan data asli sehingga dapat digunakan untuk berbagai keperluan seperti pengujian perangkat lunak atau pelatihan *machine learning.* Teknik ini juga berguna ketika data asli terbatas, sensitif atau sulit untuk didapatkan. Manfaat dari data sintesis sendiri yaitu pembuatan data tanpa batas, perlindungan privasi dan dapat mengurangi bias dalam model pelatihan AI. Aplikasi generasi data sintesis ini memiliki banyak potensi besar di berbagai bidang, terutama dalam konteks aksesibilitas data dan keamanan data.

Data sintesis dibuat dengan melibatkan penggunaan metode komputasi dan simulasi membuat data. Data yang dihasilkan ini dapat berupa teks, angka, tabel atau bahkan gambar dan video. Terdapat tiga pendekatan utama untuk menghasilkan data sintesis yaitu distribusi statistik, berbasis model dan metode deep learning. Kumpulan data sintesis dapat dibuat dengan menggunakan berbagai metode yaitu model generatif, perturbasi data, simulasi dan inferensi statistik.

Dalam artikel ini, dataset yang kami gunakan adalah data alamat (kelurahan, kecamatan, kota/kabupaten, provinsi, kode pos), *wordlist* Indonesia, nama umum orang Indonesia, layanan email, prefix tempat dan prefix administrasi. Data data tersebut dibuat dengan menggunakan beberapa fungsi seperti Library Pandas, Library Random, Fungsi Built-In Lambda, Fungsi Built-In Map dan juga Fungsi User-Defined Generate. Kombinasi dari fungsi fungsi tersebut dapat menghasilkan dataset sintesis sesuai kebutuhan analisis yang kami lakukan. Output dari program ini adalah data sintesis yang berbentuk dictionary atau JSON.

2. Metode

Pada pembuatan program aplikasi generasi data sintetis, kami menggunakan beberapa metode atau fungsi sebagai berikut:

2.1 Library Pandas

Pandas merupakan library Python yang digunakan untuk menganalisis sebuah data, dalam hal ini Pandas digunakan untuk membaca, memanipulasi dan menyimpan data dalam bentuk DataFrame.

2.2 Library Random

Dalam konteks generasi data, fungsi dalam library Random digunakan untuk menghasilkan niai acak pada variabel variabel dalam dataset.

2.3 Fungsi Built-in Lambda

Fungsi lambda digunakan untuk membuat fungsi dalam satu baris yang bersifat anonim. Dalam program ini kami menggunakan fungsi lambda untuk mendefinisikan fungsi-fungsi yang digunakan dalam pengacakan seperti acakList, acakIndex, acakHuruf, acakDaerah, acakNama, dan acakAngka.

2.4 Fungsi Built-in Map

Fungsi map digunakan untuk menerapkan suatu fungsi pada elemen dari objek yang bertipe iterable. Pada program ini kami menggunakan fungsi map untuk menerapkan fungsi 'kapital' pada elemen dalam 'namaOrang' yang diberikan dalam list.

2.5 Fungsi "User-Defined" Generate

Fungsi generate adalah fungsi yang menghasilkan data sintetis dengan cara menggabungkan hasil dari fungsi lambda dan data yang diambil. Jika dilihat pada program ini maka itu seperti nama, alamat, nomor telepon dan yang lainnya.

3. Pembahasan

Pada artikel ini kami menggunakan dataset berupa data alamat, data wordlist Indonesia, data nama umum di Indonesia, layanan email, prefix tempat dan juga prefix administrasi. Dalam pembuatannya kami menggunakan beberapa fungsi yang dikombinasikan untuk membentuk dataset tersebut, yaitu fungsi Built-In Lambda **'Lambda',** fungsi Built-In Map **'map()**', fungsi "User-Defined" **Generate()**, dan menggunakan dua library yaitu library **Pandas** dan library **Random**. Library Pandas digunakan untuk membuat DataFrame kosong dan mengisi kolom DataFrame dengan nilai nilai acak yang dihasilkan dengan menggunakan library Random.

3.1 Pemanggilan Library Pandas

```
import pandas as pd
# Data daerah
df = pd.read_csv("/content/daerah.csv")
# Data wordlist indonesia
fword = open("/content/wordlist.txt","r")
word = fword.read().split("\n")
# Data nama orang di indonesia
fNama = open("/content/nama.txt","r")
nama = fNama.read().split("\n")
# Data layanan email
email = ['@gmail.com','@yahoo.com','@outlook.com']
# Lokasi tempat
tempat = ["Perumahan","Jalan","Gang","Komplek","Gedung"]
# Administrasi daerah Indonesia
administrasi  =
["Kelurahan","Kecamatan","Kota/Kabupaten","Provinsi","Kode Pos"]
```

Pada perintah di atas terdapat kode **'import pandas as pd'** yang digunakan untuk membuat DataFrame yang kami gunakan dalam dataset ini, yaitu data daerah, data wordlist Indonesia, data nama orang, data layanan email, data lokasi tempat dan data administrasi daerah Indonesia.

Tiap-tiap DataFrame memiliki inputan yang berbeda, seperti pada data daerah dimana DataFrame diambil dari Database Kode Pos Indonesia yang disajikan dalam bentuk CSV. Data *wordlist* Indonesia dan data nama orang Indonesia diambil dari Github. Dalam dataframe wordlist dan nama orang Indonesia terdapat kode **'split("\n')'** dimana kode ini digunakan untuk memisahkan setiap string berdasarkan karakter **("\n")**. Setiap elemen

dalam daftar ini akan menjadi satu kata. Pada data layanan email terdapat list yang berisi elemen '@gmail.com, @yahoo.com, @outlook.com'. Elemen elemen tersebut mewakili tiap domain dari layanan email yang digunakan. Selanjutnya, pada data lokasi Tempat terdapat list 'tempat' yang memiliki elemen 'Perumahan, Jalan, Gang, Komplek, Gedung'. Pada DataFrame administrasi daerah Indonesia terdapat list 'administrasi' yang memiliki elemen 'Kelurahan, Kecamatan, Kota/Kabupaten, Provinsi, Kode Pos'. Setiap elemen mewakili tingkatan administrasi di Indonesia.

3.2 Fungsi Tambahan

```
# membuat kapital perkalimat
kapital = lambda x:x[0].upper()+x[1:]
# menggabungkan list
merge = lambda lst,s="":s.join([str(v) for v in lst])
```

Kode diatas mendefinisikan dua fungsi lambda yaitu **'kapital = lambda x: x[0].upper() + x[1:]'** yang mana fungsi lambda tersebut mengambil string **'x'** sebagai argumen dan kemudian mengembalikan string yang sama dengan huruf pertama yang diubah menjadi huruf kapital dengan metode **'upper()**' dan menggabungkannya dengan sisa string sehingga menghasilkan kata yang huruf depannya diubah menjadi huruf kapital. Kemudian untuk fungsi lambda **'merge = lambda lst, s="": s.join([str(v) for v in lst])'** mengambil list **'lst'** dan string opsional **'s'** sebagai argumen. Fungsi ini berguna untuk menggabungkan semua elemen dalam list untuk menjadi satu string dengan memisahkan tiap elemen dengan string **'s**'. Kedua fungsi lambda ini dapat dengan mudah mengubah string menjadi kapital perkalimatnya dan menggabungkan semua elemen dalam list menjadi satu string.

3.3 Pemanggilan Library Random dan Fungsi Lambda

```
import random as rd

acakList = lambda lst: lst[rd.randint(0,len(lst)-1)]
acakIndex = lambda lst: rd.randint(0,len(lst)-1)
acakHuruf = lambda: acakList([chr(i+65) for i in range(26)])
acakDaerah = lambda state: state.iloc[acakIndex(state)].to_list()
acakNama = lambda: acakList(nama)
acakAngka = lambda n: str(rd.randint(1,n))
gabungList = lambda lst1,lst2: list(map(lambda x: merge(x,' '),
zip(lst2, lst1)))

def generate():
  noTelepon = ["+628"]+[acakAngka(9) for _ in range(9)]
  finalTelepon = merge(noTelepon)
  namaOrang = list(map(kapital,[acakNama() for _ in range(3)]))
  finalNama = merge(namaOrang,' ')
  finalDaerah = acakDaerah(df)[::-1]
  finalDaerah = finalDaerah[1:]+[finalDaerah[0]]
  finalDaerah = merge(gabungList(finalDaerah,administrasi),', ')
   finalEmail = f"{acakList(namaOrang)}{merge([acakAngka(9) for _ in
range(3)])}{acakList(email)}"
  finalNamaJalan = f"{acakList(tempat)} {kapital(acakList(word))}"
       finalBlok    =    f"Blok    {acakHuruf()}{acakAngka(10)}    No
{acakAngka(20)}, RT {acakAngka(10)} RW {acakAngka(10)}"
  return {
      "nama":finalNama,
      "email":finalEmail,
```

```
        "telepon":finalTelepon,
        "alamat":f"{finalNamaJalan}, {finalBlok}. {finalDaerah}"
    }
```

Di dalam perintah terdapat kode **‘import random as rd’** yang mana kode ini digunakan untuk memanggil library random yang berguna untuk menghasilkan angka acak. Didalam perintah di atas terdapat beberapa fungsi lambda yang memiliki list yang berbeda beda. Kemudian terdapat fungsi **‘acakList = lambda lst: lst[rd.radint(0,len(lst)-1)]’** yang berfungsi untuk mengacak list dengan melakukan acakan berdasarkan panjang list. Output yang dihasilkan adalah isi dari list tersebut.

Pada perintah diatas, akan dijelaskan pendekatan dan langkah-langkah yang digunakan dalam aplikasi generasi data sintesis. Secara umum, aplikasi ini memanfaatkan library Python populer seperti Pandas dan random untuk memfasilitasi pengolahan data dan proses pengacakan. Pertama, data-data pendukung seperti daftar daerah, wordlist bahasa Indonesia, daftar nama orang, layanan email populer, jenis lokasi tempat, dan tingkatan administrasi daerah di Indonesia dipersiapkan. Data daerah dibaca dari file CSV menggunakan fungsi pd.read_csv dari library Pandas. Sementara itu, data wordlist, nama orang, dan lainnya dibaca dari file teks dengan metode open dan read yang disertai pemisahan berdasarkan garis baru menggunakan **‘split("\n")’**.

Selanjutnya, sejumlah fungsi lambda didefinisikan untuk membantu proses pengacakan data. Fungsi **acakList** digunakan untuk mengacak elemen dalam sebuah list secara acak. Fungsi **acakIndex** berfungsi untuk memilih indeks secara acak dari sebuah list. Fungsi **acakHuruf** menghasilkan huruf acak berdasarkan daftar huruf A-Z yang dihasilkan dari pengubahan kode Unicode ke karakter. Fungsi **acakDaerah** mengacak baris pada DataFrame daerah dan mengubahnya menjadi sebuah list. Fungsi **acakNama** mengacak nama dari daftar nama yang tersedia, sedangkan **acakAngka** menghasilkan angka acak dengan panjang tertentu dalam bentuk string.

Metode utama dalam aplikasi ini adalah fungsi **generate()**. Dalam fungsi ini, beberapa langkah dilakukan untuk menghasilkan data sintesis yang meliputi nomor telepon, nama lengkap orang, alamat daerah, alamat email, nama jalan, dan nama blok rumah/gedung. Proses dimulai dengan membangkitkan nomor telepon dengan format +628 diikuti 9 angka acak menggunakan fungsi **acakAngka**. Nama lengkap dibentuk dengan menggabungkan tiga nama acak dari daftar nama yang tersedia menggunakan fungsi **acakNama**. Alamat daerah dihasilkan dengan mengacak informasi kelurahan, kecamatan, kota/kabupaten, provinsi, dan kode pos dari data daerah yang telah dipersiapkan sebelumnya dengan memanfaatkan fungsi **acakDaerah**.

Kemudian alamat email dirangkai dengan menggabungkan nama acak, tiga angka acak yang dihasilkan **acakAngka**, dan domain email populer seperti Gmail, Yahoo, atau Outlook. Nama jalan dibuat dengan mengombinasikan jenis lokasi acak seperti perumahan, jalan, gang, dan lain-lain dengan kata acak dari wordlist menggunakan fungsi **acakList**. Terakhir, nama blok rumah atau gedung ditambahkan dalam format "Blok" diikuti huruf acak dari **acakHuruf**, angka acak dari **acakAngka**, nomor acak, RT acak, dan RW acak. Dengan memanfaatkan fungsi-fungsi tersebut serta data-data pendukung yang telah disiapkan, perintah kode tersebut mampu menghasilkan data sintesis yang realistis dan lengkap, mencakup berbagai informasi penting seperti identitas personal, alamat, kontak, dan lain sebagainya.

3.4 Penerapan Fungsi

```
total = 10
print("Total",total,"data:")
print("===========")
```

```
for i in range(total):
  print(generate())
```

Pada bagian ini, kode yang diberikan mengimplementasikan sebuah fungsi bernama **generate()** yang berfungsi untuk menghasilkan data sintesis secara otomatis. Fungsi ini memanfaatkan data-data yang telah dipersiapkan sebelumnya, seperti daftar daerah, wordlist, nama, layanan email, jenis lokasi, dan administrasi daerah.

Untuk memudahkan pengguna dalam mengatur jumlah data sintesis yang ingin dihasilkan, disediakan sebuah slider yang dapat digeser. Jumlah data yang diinginkan disimpan dalam variabel total.

Setelah jumlah data ditentukan, program akan memanggil fungsi **generate()** sebanyak total kali dalam sebuah loop. Setiap kali fungsi **generate()** dipanggil, ia akan menghasilkan satu set data sintesis yang terdiri dari nama lengkap, alamat email, nomor telepon, dan alamat lengkap yang dibentuk dari kombinasi informasi seperti jenis lokasi, nama jalan, blok rumah/gedung, kelurahan, kecamatan, kota/kabupaten, provinsi, dan kode pos.

Data sintesis yang dihasilkan kemudian dicetak ke layar dalam format yang mudah dibaca. Setiap set data sintesis ditampilkan dalam bentuk dictionary yang berisi variabel-variabel seperti 'nama', 'email', 'telepon', dan 'alamat'. Dengan menggunakan kode ini, pengguna dapat dengan mudah menghasilkan data sintesis yang realistis dan lengkap dalam jumlah yang diinginkan dengan mengubah variabel total. Data sintesis ini dapat dimanfaatkan untuk berbagai keperluan, seperti pengujian sistem, pelatihan model machine learning, atau simulasi skenario tertentu yang membutuhkan data palsu namun masuk akal.

4. Kesimpulan

Berdasarkan pembahasan yang sudah dijelaskan mengenai aplikasi generasi data sintetis menggunakan pemrograman python maka dapat disimpulkan penggunaan generasi data sintetis memiliki manfaat yang signifikan, baik pada konteks privasi dan keamanan, pembuatan data tanpa batas dan yang lainnya. Meskipun memiliki banyak manfaat, perlu dipastikan bahwa data sintetis yang dihasilkan haruslah memiliki kualitas yang memadai untuk dapat merepresentasikan data asli dengan baik.

5. Referensi

Apa itu Data Sintetis? - Penjelasan Data Sintetis. (n.d.). AWS. Retrieved March 12, 2024, from https://aws.amazon.com/id/what-is/synthetic-data/

eplusgo. (n.d.). *Database Kode Pos Indonesia*. https://www.eplusgo.com/database-kode-pos-indonesia/

Geovedi, J .(n.d.). *Indonesian Wordlist*. https://github.com/geovedi/indonesian-wordlist/blob/master/05-ivanlanin2011-sort-alpha.lst

Maulvi. (n.d.). *Nama Nama Orang Indonesia*. https://gist.github.com/maulvi/e443e22b82a1dc24e14344b47f0a80ea

Ye, Y., & Talburt, J. R. (2019, April 1). Generating synthetic data to support entity resolution education and research. *Journal of Computing Sciences in Colleges*, *34*(7), 12-19.

Link Colab:
https://colab.research.google.com/drive/1YmCRh-h9HsfGH7cjpQANKEpS6QblpnTy?usp=sharing