APLIKASI PENDETEKSI DUPLIKAT DAN PENGHAPUS DUPLIKAT PADA DATA

Natasya Ega Lina Marbun[1], Khusnun Nisa[2], Presilia[3], Irvan Alfaritzi[4], Syalaisha Andini Putriansyah[5]

Program Studi Sains Data, Fakultas Sains, Institut Teknologi Sumatera.

Email: natasya.122450024@student.itera.ac.id, khusnun.122450078@student.itera.ac.id, presilia.122450081@student.itera.ac.id, irvan.122450093@student.itera.ac.id, syalaisha.122450111@student.itera.ac.id

1. Pendahuluan

Di era digitalisasi sekarang data merupakan hal penting dan utama di berbagai bidang. Segala sistem dirancang berdasarkan data yang telah dikumpulkan dan diolah sedemikian rupa, maka dari itu penting untuk memilih dan memilah data yang akan digunakan sehingga tidak terdapat kesalahan. Salah satu kesalahan yang sering terdapat dalam suatu data adalah duplikasi data, hal tersebut merupakan fenomena dimana adanya dua atau lebih data identik dalam suatu dataset yang sama. Hal ini dapat terjadi dalam berbagai konteks seperti *database*, *spreadsheet*, *file*, dan bahkan sistem informasi yang berbeda.

Duplikasi dalam data memiliki dampak yang kurang baik seperti ruang penyimpanan menjadi boros, dapat menyebabkan inkonsistensi data dan anomali, analisis terhadap data jadi kurang akurat, juga meningkatkan resiko pelanggaran data dan akses yang tidak sah. Maka penting untuk memeriksa ada tidaknya duplikasi yang terdapat dalam data yang hendak kita gunakan.

Artikel ini akan menjelaskan penerapan dari aplikasi pendeteksi duplikasi yang merupakan solusi dari masalah tersebut. Dimana aplikasi ini menggunakan penerapan fungsi *map* dan *reduce* untuk mendeteksi dan menghapus duplikasi data dari sebuah dataset yang akan digunakan hingga diolah sedemikian rupa sehingga mendapatkan *insight* yang bermanfaat bagi khalayak ramai.

2. Metode

2.1. Fungsi *Reduce*

Fungsi *reduce* hampir sama dengan fungsi *map*() yaitu melakukan iterasi melalui elemen-elemen *iterable* yang diberikan. fungsi *map*() mengembalikan *iterable* baru yang berisi hasil

dari operasi yang diterapkan pada setiap elemen sedangakan fungsi *reduce*() akan mengembalikan nilai tunggal yang merupakan hasil akhir dari menyusutkan iterable.

2.2. Fungsi Lambda

Fungsi lambda sering disebut juga sebagai *anonymous function* karena tidak memerlukan deklarasi nama seperti fungsi biasa yang dibuat dengan def. Lambda *expression* sangat berguna ketika kita ingin membuat fungsi sederhana dalam satu baris ekspresi.

2.3. Fungsi Map

Fungsi *map*() adalah salah satu *built-in function* di Python yang dapat digunakan untuk menerapkan sebuah fungsi pada semua elemen dari objek yang bertipe *iterable* seperti list, tuple, dan sebagainya. Fungsi ini digunakan untuk memetakan suatu data layaknya fungsi peta (map).

3. Pembahasan

3.1. Kode Program

3.1.1. Impor Modul

```
1 from functools import reduce
2 import pandas as pd
3
4 def remove_duplicates(data):
5     unique_data = reduce(lambda d, x: {**d, x: None}, data, {})
6     return list(unique_data.keys())
7
```

Gambar 1. Memasukkan modul

Langkah pertama dalam membuat program yaitu memasukkan modul dan/ fungsi. Dalam program aplikasi pendeteksi duplikat, modul yang digunakan adalah modul *functools* dan *pandas*.

3.1.2. Fungsi

```
1 from functools import reduce
2 import pandas as pd
3
4 def remove_duplicates(data):
5     unique_data = reduce(lambda d, x: {**d, x: None}, data, {})
6     return list(unique_data.keys())
7
```

Gambar 2. Fungsi menghapus duplikat

Fungsi *remove_duplicates*() pada gambar 2. dibuat dengan argumen data dan variabel *unique_data* yang diinisiasi dengan list data yang telah dihapus data duplikatnya menggunakan fungsi *reduce*. *Expression* lambda pada fungsi *reduce* digunakan untuk menyimpan penambahan elemen x pada setiap iterasi ke dalam *dictionary* “d” dengan nilai *None*. Ketika fungsi *remove_duplicates*(*data*) dipanggil maka akan mengembalikan list dari kata kunci *dictionary* pada variabel *unique_data*.

```python
def read_data_from_list():
    data = input("Masukkan data list (dipisahkan koma): ")
    data_list = data.split(',')
    return [entry.strip() for entry in data_list]

def read_data_from_csv(file_path):
    try:
        data = pd.read_csv(file_path, header=None)
        return list(map(str, data.values.flatten()))
    except FileNotFoundError:
        print("File tidak ditemukan.")
        return None
```

Gambar 3. Fungsi Impor Data berbentuk list dan csv

Kode program gambar 3, pada bagian pertama menggunakan *input*, *Data.split*(‘,’), [*entry.strip*() *for entry in data list*] bertujuan untuk membaca data dari pengguna dalam bentuk *string*, memisahkan entri menggunakan koma, dan mengembalikan daftar entri yang telah dipisahkan.

Selanjutnya, pada bagian kedua terdapat *pd.read_csv*(*file_path,header=None*) bertujuan untuk membaca data dari file CSV yang tidak memiliki baris *header*. *data.values.flatten*() dan *list*(*map*(*str,data.values.flatten*())) berfungsi mengambil nilai dari *data frame* kemudian mengonversi nilai-nilai tersebut menjadi *string* dan mengembalikan dalam bentuk list, serta memberikan pesan “file tidak ditemukan.” jika file yang ditentukan tidak dapat ditemukan.

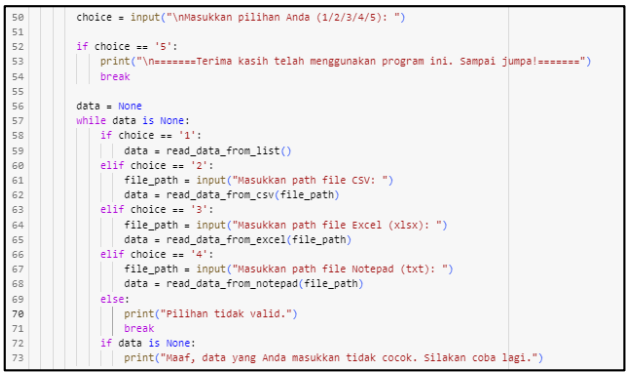
```python
def read_data_from_excel(file_path):
    try:
        data = pd.read_excel(file_path, header=None)
        data_list = data.values.flatten()
        paired_data = list(map(str, [f"{data_list[i]}, {data_list[i+1]}" for i in range(0, len(data_list), 2)]))
        return paired_data
    except FileNotFoundError:
        print("File tidak ditemukan.")
        return None

def read_data_from_notepad(file_path):
    try:
        with open(file_path, 'r') as file:
            data = file.readlines()
        return list(map(str.strip, data))
    except FileNotFoundError:
        print("File tidak ditemukan.")
        return None
```

Gambar 4. Fungsi Impor Data berbentuk excel dan txt

Kode program pada gambar 4, memiliki 2 fungsi tambahan. Pada bagian pertama pd.read_excel(*file_path, header=None*) bertujuan untuk membaca data file dari file excel menggunakan *pandas* dengan file excel tersebut tidak memiliki baris *header*. [*f*’{*data_list*[i]}, …, *len*(*data_list,* 2], *list*(*map*(*str*,...)) dan *Return paired_data* berfungsi membuat pasangan nilai dengan menggabungkan setiap dua nilai berturut-turut, dan kemudian mengonversi hasilnya menjadi *string* dan mengembalikan data hasil pembacaan dalam bentuk list pasangan nilai.

Selanjutnya, pada bagian kedua bertujuan membaca data dari file teks(Notepad). *With open*(*file_path*, ‘r’) as file berfungsi untuk membuka file teks dalam mode baca. *file.readlines*() berfungsi setiap baris dalam file dan menyimpannya sebagai list. *list*(*map*(*str.strip,data*)) sebagai penghapus spasi tambahan dari setiap baris menggunakan metode *strip*(). *Return data* berfungsi mengembalikan data hasil pembacaan dalam bentuk list.



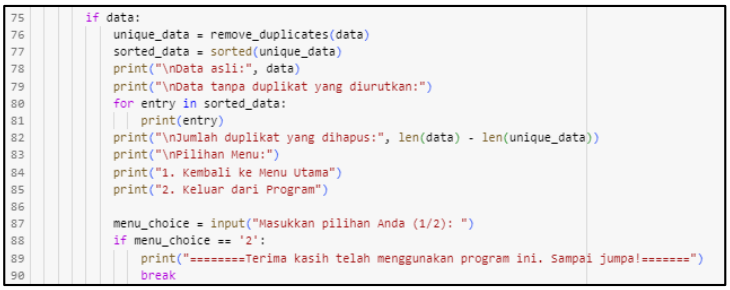
```
        choice = input("\nMasukkan pilihan Anda (1/2/3/4/5): ")

        if choice == '5':
            print("\n=======Terima kasih telah menggunakan program ini. Sampai jumpa!=======")
            break

        data = None
        while data is None:
            if choice == '1':
                data = read_data_from_list()
            elif choice == '2':
                file_path = input("Masukkan path file CSV: ")
                data = read_data_from_csv(file_path)
            elif choice == '3':
                file_path = input("Masukkan path file Excel (xlsx): ")
                data = read_data_from_excel(file_path)
            elif choice == '4':
                file_path = input("Masukkan path file Notepad (txt): ")
                data = read_data_from_notepad(file_path)
            else:
                print("Pilihan tidak valid.")
                break
            if data is None:
                print("Maaf, data yang Anda masukkan tidak cocok. Silakan coba lagi.")
```

Gambar 5. Menampilkan Menu

Fungsi *main*() pada gambar 5. dibuat sebagai program utama dari aplikasi pendeteksi duplikat. Pada Fungsi *main*() terdapat *variabel choice* yang akan diinputkan oleh pengguna. Variabel choices menyimpan nilai dari jenis data yang dipilih untuk diolah. Selanjutnya terdapat kondisi percabangan dimana program akan berjalan sesuai dengan nilai *variabel choice* yang telah diinputkan oleh pengguna.



```
    if data:
        unique_data = remove_duplicates(data)
        sorted_data = sorted(unique_data)
        print("\nData asli:", data)
        print("\nData tanpa duplikat yang diurutkan:")
        for entry in sorted_data:
            print(entry)
        print("\nJumlah duplikat yang dihapus:", len(data) - len(unique_data))
        print("\nPilihan Menu:")
        print("1. Kembali ke Menu Utama")
        print("2. Keluar dari Program")

        menu_choice = input("Masukkan pilihan Anda (1/2): ")
        if menu_choice == '2':
            print("========Terima kasih telah menggunakan program ini. Sampai jumpa!=======")
            break
```

Gambar 6. Fungsi mengurutkan, menghapus dan menampilkan data

Kode program pada gambar 6. merupakan bagian dari fungsi *main*() dimana saat pengguna telah memasukkan data maka data yang duplikat akan dihapus menggunakan fungsi *remove_duplicates*(data) dan disimpan kedalam variabel *unique_data*. Kemudian setelah proses penghapusan data duplikat selesai maka pengguna diminta untuk melanjutkan atau keluar dari program.

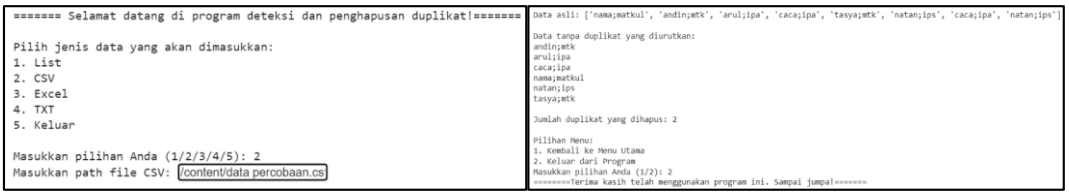
```
if __name__ == "__main__":
    main()
```

Gambar 7. Tempat Data disimpan

Kondisi percabangan pada gambar 7. Merupakan sintaksis dalam bahasa pemrograman Python yang umum digunakan untuk memastikan bahwa suatu skrip Python hanya dijalankan jika dieksekusi secara langsung (bukan diimpor sebagai modul). Dengan demikian, jika skrip Python dijalankan secara langsung, fungsi *main*() akan dieksekusi.

3.2. Hasil Program

Tampilan dari program ini dapat dilihat pada gambar 8 berikut ini.



Gambar 8. Tampilan Menu Aplikasi dan Hasil Data yang telah di diteksi, dihapus, dan diurutkan

4. Kesimpulan

Aplikasi pendeteksi duplikat dan penghapus duplikat yang diimplementasikan pada kode program tersebut memberikan solusi efektif untuk mengelola dan membersihkan data ganda. Dengan dukungan fungsionalitas seperti membaca data dari berbagai sumber, termasuk list, file CSV, file Excel, file teks (Notepad), dan data lainnya dari berbagai format. Kemampuannya dalam mendeteksi dan menghapus duplikat dengan menggunakan metode penghapusan yang efisien, seperti reduksi dengan fungsi lambda, membuatnya efektif untuk menangani jumlah data yang besar tanpa mengorbankan kinerja.

Selain itu, aplikasi ini juga memberikan pengalaman pengguna yang ramah dengan menyediakan menu interaktif yang mudah dimengerti. Pengguna dapat dengan mudah memilih jenis data yang akan dimasukkan, baik itu dari list manual, file CSV, Excel, atau Notepad. Hasil penghapusan duplikat yang diurutkan dan ditampilkan dengan jelas memberikan gambaran yang bersih dan terstruktur dari data yang telah diproses. Keseluruhan, aplikasi ini memberikan solusi yang dapat diandalkan untuk mengatasi masalah duplikat dalam data, memberikan pengguna kemampuan untuk dengan cepat membersihkan dan menganalisis data mereka.

5. Referensi

[1] F. L. d. H. P. Ningrum, "High Order Function for Processing Structured Data," in *Modul 3*, Malang, Universitas Muhammadiyah Malang, 2023.

[2] A. Muhardin, "Belajar Python: Membuat Fungsi dengan Lambda Expression," petanikode, 18 December 2019. [Online]. Available: https://www.petanikode.com/python-lambda/. [Accessed 11 March 2024].

[3] P. Sanbersy, "Function and Methods," PT Sanbersy, 19 November 2021. [Online]. Available: https://blog.sanbercode.com/docs/data-science-versi-3/pekan-1/function-and-methods/. [Accessed 11 March 2024].

[4] Alza, "Belajar koding untuk pemula," alza.web.id, 22 June 2019. [Online]. Available: https://koding.alza.web.id/menggunakan-reduce-functools/. [Accessed 11 March 2024].