

# APLIKASI NORMALISASI DATA

**Arafi Ramadhan Maulana<sup>1)</sup>, Dwi Ratna Anggraeni<sup>2)</sup>, Raid Muhammad Naufal<sup>3)</sup>, Rayan Koemi Karuby<sup>4)</sup>, Muhammad Deriansyah Okutra<sup>5)</sup>**

Program Studi Sains Data, Fakultas Sains, Institut Teknologi Sumatera

Email : [arafi.122450002@student.itera.ac.id](mailto:arafi.122450002@student.itera.ac.id)<sup>1</sup>, [dwi.122450008@student.itera.ac.id](mailto:dwi.122450008@student.itera.ac.id)<sup>2</sup>,  
[raid.122450027@student.itera.ac.id](mailto:raid.122450027@student.itera.ac.id)<sup>3</sup>, [rayan.122450038@student.itera.ac.id](mailto:rayan.122450038@student.itera.ac.id)<sup>4</sup>,  
[Muhhammad.122450101@student.itera.ac.id](mailto:Muhhammad.122450101@student.itera.ac.id)<sup>5</sup>

## 1. Pendahuluan

Di era saat ini, semua perusahaan baik *startup* maupun yang berskala besar mengumpulkan dan mengolah data agar dapat digunakan untuk mengembangkan perusahaan mereka. salah satu cara yang digunakan dalam mengelola data adalah normalisasi data. Dimana normalisasi data adalah cara merubah nilai numerik dalam sebuah dataset menjadi skala umum, dengan tidak mendistorsi perbedaan dalam interval nilai.

Normalisasi data memudahkan kita dalam memilah dan menyortir informasi yang ada di dalam data, saat memodifikasi data kita dapat mencegah masalah. Contoh metode yang digunakan dalam normalisasi data biasanya yaitu mengubah data dari rentang asli menjadi skala tertentu, misalnya skala 0-1 disebut dengan Normalisasi *Min-Max*, atau dengan mengurangi nilai rata-ratanya.

Dalam bahasan kali ini kita menggunakan bahasa pemrograman Python untuk Aplikasi Normalisasi Data dengan membuat fungsi yang menggunakan map dan lambda. Fungsi map merupakan fungsi yang dapat menerapkan sebuah fungsi pada seluruh elemen baik dari data berbentuk list, tuple, dan sebagainya. Sedangkan lambda adalah *anonymous function* atau fungsi tanpa nama yang memungkinkan kita membuat fungsi dalam satu baris yang biasanya disimpan dalam sebuah variabel. Dengan menerapkan fungsi-fungsi tersebut kita dapat dengan mudah melakukan perubahan matematis di setiap elemen data tanpa menuliskan perulangan secara eksplisit.

Pada jurnal ini, kita akan mengeksplor penggunaan fungsi lambda dan map dalam membuat Aplikasi Normalisasi Data. Penggunaan fungsi-fungsi tersebut dapat lebih efektif dan fleksibel dalam analisis data. Normalisasi data bertujuan untuk menyederhanakan dan mempersiapkan data agar dapat diinterpretasikan dan dimanfaatkan secara baik. Aplikasi normalisasi data kali ini kita terapkan pada data inflasi Indonesia yang bersumber dari kaggle. Dengan melakukan normalisasi pada data inflasi, kita dapat meningkatkan keakuratan model yang dibuat untuk membantu memperkirakan atau menjelaskan inflasi di masa yang akan datang.

## 2. Metode

### 2.1 Penggunaan Numpy

Salah satu pustaka Python yang digunakan secara berkali-kali adalah Numpy. Numpy merupakan salah satu dari sekian banyak pustaka Python yang fokusnya adalah untuk melakukan perhitungan pada *array*, dan melakukan perhitungan

matematika dan statistika dasar pada data dalam kode. Dalam kode program aplikasi kali ini, Numpy difokuskan untuk membantu dalam perhitungan dasar statistika untuk mencari komponen-komponen yang dibutuhkan untuk melakukan normalisasi.

## 2.3 Pandas

Pandas adalah suatu pustaka dalam Python yang seringkali digunakan untuk melakukan manipulasi pada suatu *dataset*. Pandas dapat membantu dalam memberikan struktur data yang kuat pada *dataset* atau suatu *DataFrame*. Pada kode pemrograman aplikasi kali ini, Pandas digunakan untuk *import dataset* yang akan digunakan dan elemennya dinormalisasi.

## 2.4 Lambda

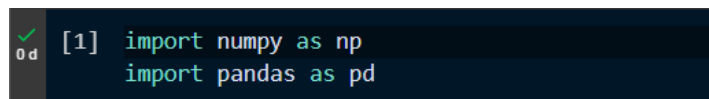
Lambda merupakan suatu fitur dalam bahasa pemrograman Python yang digunakan untuk membuat fungsi tanpa harus dinamakan. Keunggulan dalam menggunakan lambda adalah lambda dapat mendefinisikan beberapa baris kode menjadi hanya satu baris kode, Lambda pada program kali ini digunakan untuk meringkas baris kode dan elemen data yang merupakan bagian dari *mapping*. Hal ini memudahkan untuk melakukan normalisasi tanpa harus menulis fungsi secara terpisah.

## 2.5 Map

Fungsi `map()` merupakan salah satu fungsi bawaan (*built in function*) Python yang membantu dalam menerapkan fungsi tertentu ke dalam setiap elemen dari suatu data atau list dan di-*return* dalam bentuk suatu *iterator*. Dalam kode program ini diterapkan fungsi `map()` dalam penggunaannya pada fungsi dari normalisasi yang sudah dibuat agar memudahkan untuk memetakan fungsinya ke dalam *array* yang ada.

# 3. Pembahasan

## 3.1 Import Library



```
[1] import numpy as np
import pandas as pd
```

Gambar 1. Import *Library*

Pada bagian ini, kita mengimport library numpy dan pandas, kedua library ini biasanya digunakan untuk komputasi numerik dan memanipulasi data.

## 3.2 Import Dataset

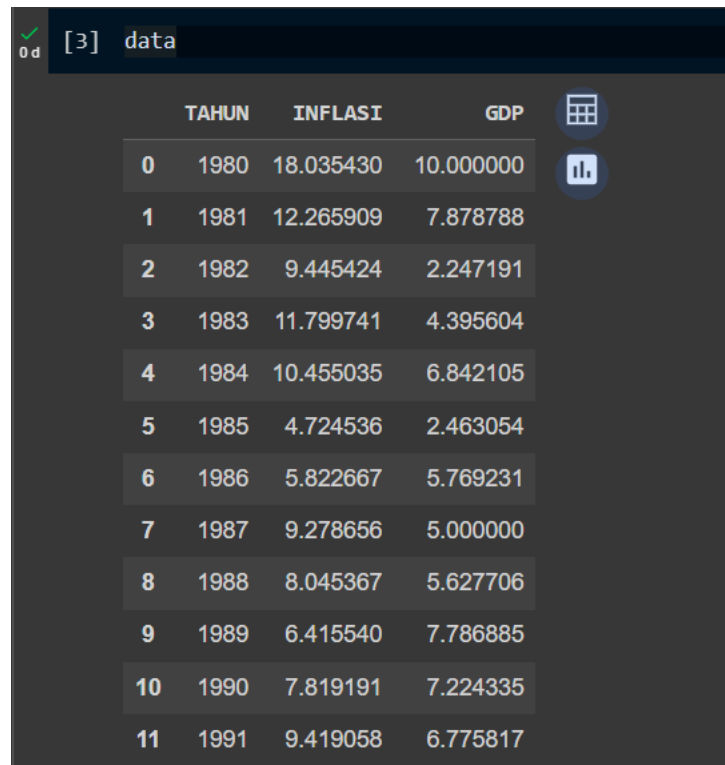


```
[2] data = pd.read_csv("/content/Indonesia.csv")
```

Gambar 2. Import *Dataset*

Pada bagian ini, untuk mengimport *dataset* ke Google Colab digunakan format kode `data = pd.read_csv()`, agar lebih mudah dipanggilnya.

### 3.3 Menampilkan Dataset



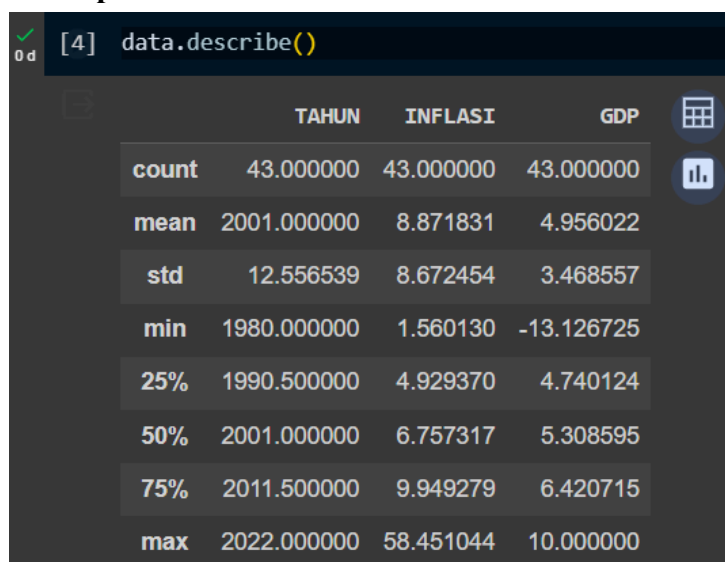
The screenshot shows a Jupyter Notebook cell with the code `[3] data` executed. The output is a table with 12 rows and 3 columns: TAHUN, INFLASI, and GDP. The rows are indexed from 0 to 11.

	TAHUN	INFLASI	GDP
0	1980	18.035430	10.000000
1	1981	12.265909	7.878788
2	1982	9.445424	2.247191
3	1983	11.799741	4.395604
4	1984	10.455035	6.842105
5	1985	4.724536	2.463054
6	1986	5.822667	5.769231
7	1987	9.278656	5.000000
8	1988	8.045367	5.627706
9	1989	6.415540	7.786885
10	1990	7.819191	7.224335
11	1991	9.419058	6.775817

Gambar 3. Tampilan *Dataset*

Ini adalah *dataset* yang kami gunakan, bisa dilihat dari kode sebelumnya dimana *dataset* tersebut disimpan pada variabel `data` sehingga pada saat memanggil `data` maka akan menampilkan *output* berupa *dataset*.

### 3.4 Statistika Deskriptif



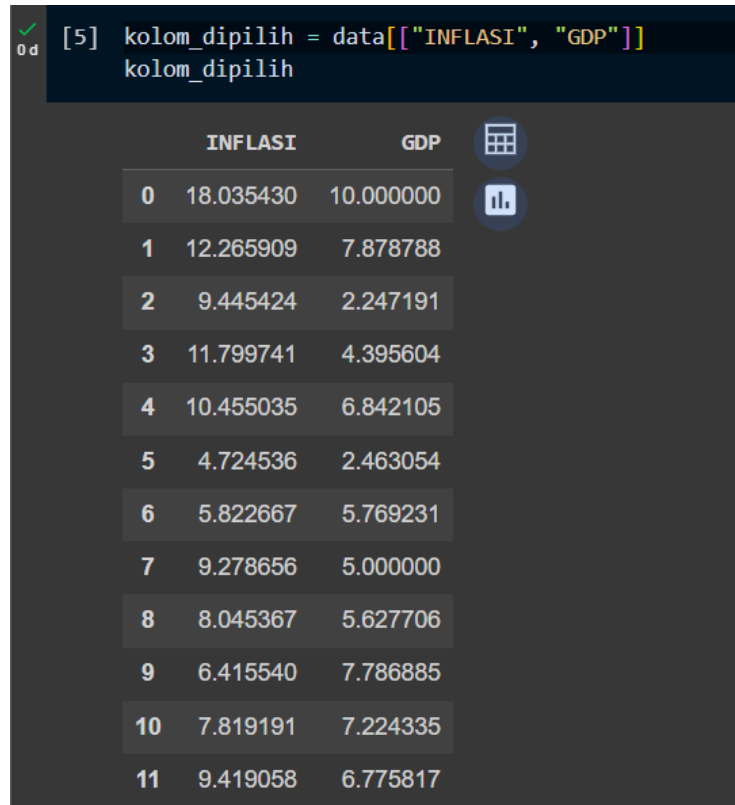
The screenshot shows a Jupyter Notebook cell with the code `[4] data.describe()` executed. The output is a table with 9 rows and 4 columns: the first column contains statistical measures, and the next three columns contain values for TAHUN, INFLASI, and GDP.

	TAHUN	INFLASI	GDP
count	43.000000	43.000000	43.000000
mean	2001.000000	8.871831	4.956022
std	12.556539	8.672454	3.468557
min	1980.000000	1.560130	-13.126725
25%	1990.500000	4.929370	4.740124
50%	2001.000000	6.757317	5.308595
75%	2011.500000	9.949279	6.420715
max	2022.000000	58.451044	10.000000

Gambar 4. Statistika Deskriptif

Pada bagian ini kami menggunakan kode “data.describe()”, untuk mengetahui statistika deskriptif dari data yang kami gunakan, deskripsinya berupa *count* (banyak data), *mean* (rata-rata), *std* (standar deviasi), *min* (nilai terkecil), *max* (nilai terbesar), dan lainnya.

### 3.5 Kolom Data Normalisasi



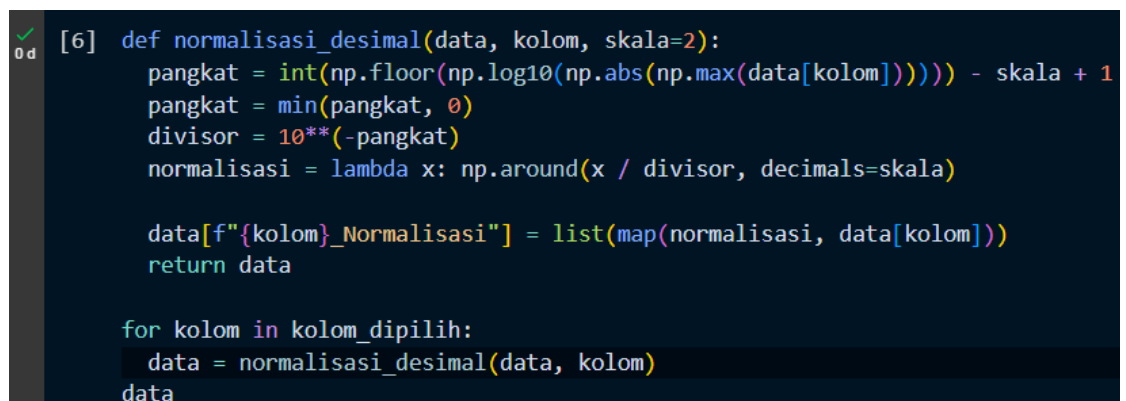
```
[5] kolom_dipilih = data[['INFLASI', 'GDP']]
kolom_dipilih
```

	INFLASI	GDP
0	18.035430	10.000000
1	12.265909	7.878788
2	9.445424	2.247191
3	11.799741	4.395604
4	10.455035	6.842105
5	4.724536	2.463054
6	5.822667	5.769231
7	9.278656	5.000000
8	8.045367	5.627706
9	6.415540	7.786885
10	7.819191	7.224335
11	9.419058	6.775817

Gambar 5. Kolom untuk Normalisasi

Pada bagian ini kami menginisialisasi “kolom\_dipilih” sebagai kolom “INFLASI” dan “GDP” pada *dataset* untuk dinormalisasi yang merupakan tipe data numerik dan lalu menampilkan “kolom\_dipilih”.

### 3.6 Normalisasi Data



```
[6] def normalisasi_desimal(data, kolom, skala=2):
    pangkat = int(np.floor(np.log10(np.abs(np.max(data[kolom]))))) - skala + 1
    pangkat = min(pangkat, 0)
    divisor = 10**(-pangkat)
    normalisasi = lambda x: np.around(x / divisor, decimals=skala)

    data[f"{kolom}_Normalisasi"] = list(map(normalisasi, data[kolom]))
    return data

for kolom in kolom_dipilih:
    data = normalisasi_desimal(data, kolom)
data
```

	TAHUN	INFLASI	GDP	INFLASI_Normalisasi	GDP_Normalisasi
0	1980	18.035430	10.000000	18.04	10.00
1	1981	12.265909	7.878788	12.27	7.88
2	1982	9.445424	2.247191	9.45	2.25
3	1983	11.799741	4.395604	11.80	4.40
4	1984	10.455035	6.842105	10.46	6.84
5	1985	4.724536	2.463054	4.72	2.46
6	1986	5.822667	5.769231	5.82	5.77
7	1987	9.278656	5.000000	9.28	5.00
8	1988	8.045367	5.627706	8.05	5.63
9	1989	6.415540	7.786885	6.42	7.79

Gambar 6. Aplikasi Normalisasi Data

Pada bagian ini di baris pertama mendefinisikan fungsi normalisasi yang didalamnya terdapat 3 parameter, parameter data berisikan *DataFrame* yang berisi data untuk dinormalisasi, lalu pada parameter kolom untuk menormalisasi kolom pada *DataFrame*, lalu parameter skala adalah parameter opsional yang menentukan jumlah digit desimal yang diinginkan untuk dinormalisasi. Lalu pada baris 2-4 adalah fungsi untuk menghitung pangkat dari 10 yang diperlukan untuk membagi nilai-nilai dalam kolom. Pada baris 5 menampilkan cara untuk menormalisasi dengan fungsi lambda. Pada baris ke 7 untuk penerapan hasil dari normalisasi ke *DataFrame*, yang dimana menambahkan kolom baru. Lalu pada fungsi terakhir adalah untuk menampilkan semua tabel yang telah dinormalisasi.

#### 4. Kesimpulan

Dari *dataset* yang didapat data terkait hubungan antara inflasi dan GDP, pada data tersebut terdapat nilai desimal yang belum dibulatkan, karena nilai desimal tersebut mempengaruhi tahap selanjutnya yaitu analisis data, maka sebelum menganalisis diperlukannya normalisasi data, yang bertujuan untuk mengubah nilai pada *dataset* agar dapat membulatkan dan mengambil dua angka di belakang koma sehingga dapat mempermudah untuk menganalisis data. Pada normalisasi data tidak hanya dapat dilakukan dengan membulatkan nilai desimal, normalisasi data juga dapat untuk memperkecil *range* pada data sehingga mempermudah kita untuk menganalisis data.