

**MODEL PEMBELAJARAN DAN LAPORAN
AKHIR
PROJECT-BASED LEARNING
MATA KULIAH DATA MINING I
KELAS A**



**“KLASTERISASI PENYAKIT HIV DI PROVINSI JAWA BARAT
DENGAN MENGGUNAKAN METODE K-MEANS”**

DISUSUN OLEH KELOMPOK “VII” :

- | | |
|---------------------------------|---------------------------|
| 1. Ahmad Wafi Fathurrahman | (21083010011) - KETUA |
| 2. M. Nizar Riswanda | (21083010015) - ANGGOTA |
| 3. Farhan Dequika Pradana P. A. | (21083010062) - ANGGOTA |
| 4. Yayang Dimas Saputra | (21083010102) - ANGGOTA |
| 5. Valentino Belardo Sitanggang | (21083010109) - ANGGOTA |
| 6. M. Awaluddin Ilham Zuhri | (21083010112) - ANGGOTA |

DOSEN PENGAMPU:

TRESNA MAULANA FAHRUDIN, S.ST., MT (20219930501200)

PROGRAM STUDI SAINS DATA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN”
JAWA TIMUR
2023

DAFTAR ISI

DAFTAR ISI.....	3
DAFTAR GAMBAR.....	4
DAFTAR TABEL.....	5
BAB I: PENDAHULUAN.....	6
1.1. LATAR BELAKANG.....	6
1.2. PERMASALAHAN.....	6
1.3. TUJUAN.....	7
1.4. MANFAAT.....	7
BAB II: TINJAUAN PUSTAKA.....	8
2.1 TEORI PENUNJANG.....	8
2.1.1 Penyakit HIV.....	8
2.1.2 Data Mining.....	8
2.1.3 Clustering.....	9
2.1.4 Algoritma K-Means.....	9
2.2 PENELITIAN TERKAIT.....	11
BAB III: METODOLOGI PENELITIAN.....	13
BAB IV: HASIL DAN PEMBAHASAN.....	16
BAB V: KESIMPULAN.....	17
DAFTAR PUSTAKA.....	18
LAMPIRAN.....	19

DAFTAR GAMBAR

DAFTAR TABEL

BAB I: PENDAHULUAN

1.1 LATAR BELAKANG

HIV (Human Immunodeficiency Virus) adalah virus yang menyerang sistem kekebalan tubuh dan dapat menurunkan kemampuan tubuh untuk melawan infeksi dan penyakit. Padahal AIDS adalah penyakit dimana HIV sudah dalam tahap akhir infeksi. Pada tahap ini kemampuan tubuh melawan infeksi sangat lemah sehingga sangat rentan terhadap berbagai infeksi. Algoritma K-Means digunakan sebagai metode clustering penyebaran HIV/AIDS di Jawa Barat [2].

Algoritma K-Means adalah salah satu algoritma pengelompokan yang paling umum digunakan karena kesederhanaan dan efisiensi partisi yang membagi data ke dalam kelompok yang berbeda. Algoritma K-Means merupakan algoritma yang dapat meminimalkan jarak antara data dan cluster. Clustering merupakan metode yang cukup populer terutama digunakan dalam pengolahan data karena kesederhanaannya dan sangat efektif untuk mengelompokkan data dalam jumlah besar. K-Means adalah metode pengelompokan non-hierarkis yang mencoba membagi data yang ada menjadi satu atau lebih kelompok. Penggunaan algoritma K-Means dalam klasterisasi penyakit HIV/AIDS di Provinsi Jawa Barat diharapkan dapat membantu mengidentifikasi pola penyebaran penyakit tersebut.

Oleh karena itu, pada proyek ini kami bertujuan untuk melakukan klasterisasi penyakit HIV di Provinsi Jawa Barat dengan menggunakan metode K-means. Tujuan kami adalah untuk mengidentifikasi pola dan kelompok penyakit HIV berdasarkan karakteristik yang relevan. Dengan menggunakan metode ini, kami berharap dapat memperoleh pemahaman yang lebih mendalam tentang distribusi dan perubahan penyakit HIV di wilayah tersebut.

1.2 Permasalahan

1. Belum tersedianya suatu analisis yang memberikan informasi terkait klaster penyakit HIV di wilayah Jawa Barat dengan pemrograman python.

1.3 TUJUAN

1. Menyediakan suatu analisis yang memberikan informasi terkait klaster penyakit HIV di wilayah Jawa Barat dengan pemrograman python.

1.4 MANFAAT

1. Bagi Pengembangan Ilmu

Artikel ini dapat memberikan informasi yang bermanfaat tentang penyakit HIV di Provinsi Jawa Barat, melalui analisis klasterisasi ini kita dapat mengungkap pola-pola tertentu dalam data yang berkaitan dengan penyakit HIV, seperti distribusi geografis, karakteristik pasien, atau faktor risiko yang terkait. dengan demikian artikel ini dapat memperluas dan mengembangkan pengetahuan kita tentang penyakit HIV secara spesifik di suatu wilayah tersebut lalu Artikel ini juga dapat mendorong peneliti lain untuk menggunakan metode klasterisasi dalam analisis data epidemiologi mereka, karena artikel ini menunjukkan salah satu cara di mana metode klasterisasi, seperti k-means, dapat diterapkan dalam studi epidemiologi

2. Bagi Masyarakat

Artikel ini dapat memberikan pemahaman yang lebih baik kepada masyarakat tentang penyakit HIV, karena masyarakat akan mudah memahami pola klasterisasi penyakit HIV seperti faktor resiko, distribusi geografis, dan karakteristik populasi yang rentan terhadap HIV. pemahaman ini akan meningkatkan kesadaran masyarakat tentang penyakit HIV.

BAB II: TINJAUAN PUSTAKA

2.1 TEORI PENUNJANG

2.1.1 Penyakit HIV

HIV(Human Immunodeficiency Virus) adalah salah satu jenis virus yang dapat menyebabkan penyakit serius bagi penderitanya. Lantaran, HIV adalah virus yang menyerang sistem kekebalan tubuh manusia.

HIV ini menyerang salah satu sel di dalam sel darah putih, yaitu sel T atau CD4. Di mana, sel tersebut memiliki peran penting untuk menjaga imun tubuh dan memerangi infeksi yang masuk ke dalam tubuh.

Apabila tidak ditangani sesegera mungkin, infeksi HIV dapat berkembang hingga mencapai stadium akhir. Stadium akhir dari HIV adalah AIDS. AIDS atau Acquired Immune Deficiency Syndrome merupakan kondisi ketika sistem kekebalan tubuh sudah tidak mampu lagi melawan infeksi yang masuk [5].

2.1.2 Data Mining

Data mining sering disebut juga Knowledge Discovery in Database (KDD) yang merupakan proses pengumpulan dan pengolahan data yang bertujuan untuk mengekstrak informasi penting pada data. Proses pengumpulan dan ekstraksi informasi tersebut dapat dilakukan menggunakan perangkat lunak dengan bantuan perhitungan statistika, matematika, ataupun teknologi Artificial Intelligence (AI).

Metode data mining

Secara umum, terdapat beberapa metode yang digunakan untuk melakukan data mining. Berikut ini adalah metodenya:

1. Association

Teknik yang pertama adalah association. Association adalah metode berbasis aturan yang digunakan untuk menemukan asosiasi dan hubungan variabel dalam satu set data. Biasanya analisis ini terdiri dari pernyataan “if atau then” sederhana. Association banyak digunakan dalam mengidentifikasi korelasi produk dalam keranjang belanja untuk memahami kebiasaan konsumsi pelanggan. Sehingga, perusahaan dapat mengembangkan strategi penjualan dan membuat sistem rekomendasi yang lebih baik.

2. Classification

Selanjutnya classification, ia adalah metode yang paling umum digunakan dalam data mining. Classification adalah tindakan untuk memprediksi kelas suatu objek.

3. Regression

Regression adalah teknik yang menjelaskan variabel dependen melalui proses analisis variabel independen. Sebagai contoh, prediksi penjualan suatu produk berdasarkan korelasi antara harga produk dengan tingkat pendapatan rata-rata pelanggan.

4. Clustering

Terakhir, metode clustering. Clustering digunakan dalam membagi kumpulan data menjadi beberapa kelompok berdasarkan kemiripan atribut yang dimiliki. Contoh kasusnya adalah Customer Segmentation. Ia membagi pelanggan ke dalam beberapa grup berdasarkan tingkat kemiripannya.

2.1.3 Clustering

Clustering adalah salah satu teknik dari algoritma machine learning yaitu unsupervised learning. Algoritma clustering membagi populasi atau data point dengan sifat yang sama ke beberapa kelompok kecil untuk dikelompokkan. Menurut Tan (2006), clustering adalah sebuah proses untuk mengelompokkan data ke dalam beberapa cluster atau kelompok sehingga data dalam satu cluster memiliki tingkat kemiripan yang maksimum dan data antar cluster memiliki kemiripan yang minimum .

2.1.4 Algoritma K-Means

Algoritma K-means merupakan salah satu algoritma yang bersifat unsupervised learning. K-Means memiliki fungsi untuk mengelompokkan data ke dalam data cluster. Algoritma ini dapat menerima data tanpa ada label kategori. Algoritma K-means clustering mengelompokkan data berdasarkan jarak antara data terhadap titik centroid cluster yang didapatkan melalui proses berulang. Analisis perlu menentukan jumlah K sebagai input algoritma.

Adapun kelebihan dan kekurangan dari metode K-Means sebagai berikut:

a. Kelebihan Metode K-Means:

Metode K-Means memiliki beberapa kelebihan yang membuatnya populer dalam pengelompokan data, yaitu :

- Relatif sederhana dan mudah untuk diterapkan.
- Dapat diskalakan untuk dataset dalam jumlah besar.

- Mudah beradaptasi dengan contoh baru.
 - Umum diimplementasikan ke cluster dengan bentuk dan ukuran yang berbeda, seperti cluster elips.
- b. Kekurangan Metode K-Means:

Meskipun memiliki banyak kelebihan, metode K-Means juga memiliki beberapa kelemahan yang perlu diperhatikan:

- Perlu menentukan nilai k secara manual
- Sangat bergantung pada inisialisasi awal. Jika nilai random untuk inisialisasi kurang baik, maka pengelompokan yang dihasilkan pun menjadi kurang optimal.
- Dapat terjadi curse of dimensionality. Masalah ini timbul jika dataset memiliki dimensi yang sangat tinggi. Cara kerja algoritma ini adalah mencari jarak terdekat antara k buah titik dengan titik lainnya. Mencari jarak antar titik pada 2 dimensi, kemungkinan masih mudah dilakukan. Namun apabila dimensi bertambah menjadi 20 tentunya hal ini akan menjadi sulit.
- K-means mengalami kesulitan mengelompokkan data di mana cluster memiliki ukuran dan kepadatan yang bervariasi.

2.2 PENELITIAN TERKAIT

Dalam penyusunan laporan penelitian ini, kami terinspirasi dan mereferensi dari penelitian-penelitian sebelumnya yang berkaitan dengan latar belakang masalah pada penelitian ini, Berikut ini penelitian terdahulu yang berhubungan dengan laporan penelitian ini antara lain:

2.2.1 Klasterisasi Penyakit HIV/AIDS di Jawa Barat Menggunakan Algoritma K-Means Clustering

Penelitian yang dilakukan oleh Wina Rosida & Yudhistira Arie Wijaya, 2023, “Klasterisasi Penyakit HIV/AIDS di Jawa Barat Menggunakan Algoritma K-Means Clustering”(Rosida & Wijaya, 2023). Memiliki kasus yang sama dengan penelitian kami. Namun, yang membedakan adalah software yang digunakan, dimana dalam penelitian terdahulu ini menggunakan metode K-Mean Clustering

dengan software RapidMiner yang memiliki nilai DBI (Davies-Bouldin Index) terkecil 0,414 pada k-2 yang merupakan hasil cluster terbaik [2].

2.2.2 Implementasi algoritma K-Means dalam klasterisasi penyakit HIV/AIDS di Indonesia

Penelitian yang berjudul "Implementasi algoritma K-Means dalam klasterisasi penyakit HIV/AIDS di Indonesia" oleh Tita Puspita Sari, April Lia Hananto Elfina Novalia, Tukino, dan Shofa Shofia Hilabi mengusulkan solusi untuk menganalisis penyebaran penyakit HIV/AIDS di Indonesia. Penelitian ini melibatkan persiapan data jumlah penderita HIV/AIDS di berbagai wilayah Indonesia, preprocessing data, menentukan jumlah klaster yang optimal, dan menggunakan metode evaluasi seperti SSE atau Silhouette Coefficient.

Penelitian ini dapat memberikan pemahaman yang lebih baik tentang pola dan karakteristik penyebaran penyakit HIV/AIDS di berbagai wilayah, dan dapat digunakan sebagai dasar untuk pengambilan keputusan dalam penanggulangan dan pencegahan penyakit tersebut di Indonesia. Selain itu, penelitian ini juga mengimplementasikan metode pengelompokan data K-Means Clustering dalam bentuk aplikasi berbasis web menggunakan bahasa pemrograman PHP dan MySQL sebagai sumber penyimpanan data [3].

2.2.3 Penerapan Algoritma K-Means Clustering Analysis Pada Kasus Penderita HIV/AIDS (Studi Kasus Kabupaten Banjar)

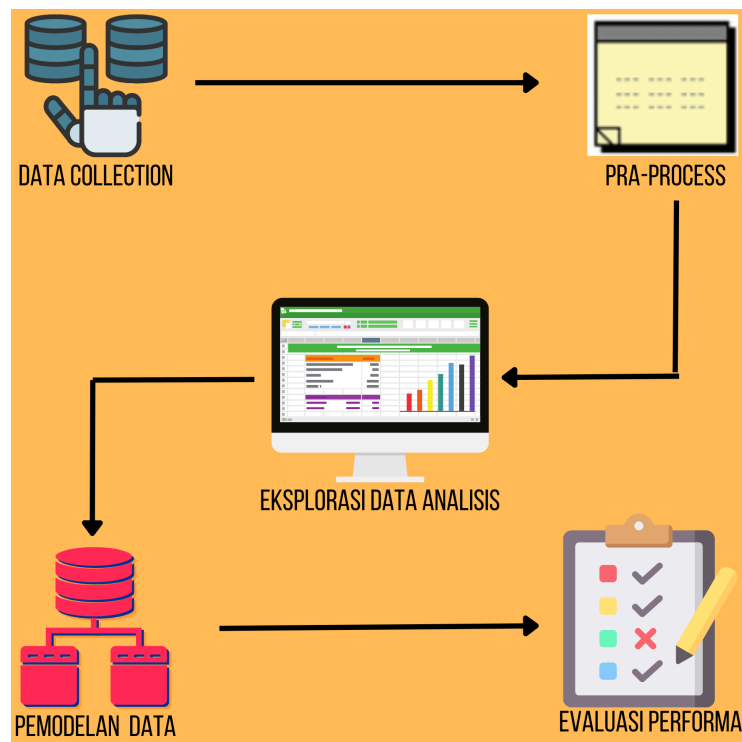
Penelitian selanjutnya yang berjudul "Penerapan Algoritma K-Means Clustering Analysis Pada Kasus Penderita HIV/AIDS (Studi Kasus Kabupaten Banjar)" yang ditulis oleh Hayati Noor, Adani Dharmawati, dan Tri Wahyu Qur'ana bertujuan untuk mengatasi masalah penyebaran penyakit HIV/AIDS di Kabupaten Banjar dengan menggunakan algoritma K-Means Clustering. Dalam penelitian ini, data penderita HIV/AIDS di Kabupaten Banjar dikumpulkan, termasuk informasi seperti usia, jenis kelamin, status perkawinan, tingkat pendidikan, dan faktor risiko lainnya. Langkah-langkah solusi yang dilakukan meliputi preprocessing data, menentukan jumlah klaster yang optimal, dan mengimplementasikan algoritma K-Means pada data tersebut.

Hasil klasterisasi dianalisis untuk mengidentifikasi pola dan karakteristik penyebaran penyakit HIV/AIDS di Kabupaten Banjar, serta menghubungkannya

dengan faktor risiko yang terkait. Penelitian ini diharapkan dapat memberikan pemahaman yang lebih baik tentang pola penyebaran penyakit HIV/AIDS di wilayah tersebut, yang dapat menjadi dasar untuk pengambilan keputusan dalam upaya penanggulangan dan pencegahan penyakit tersebut di Kabupaten Banjar. Metode yang digunakan yaitu K-Means Clustering dengan software RapidMiner [4].

BAB III: METODOLOGI PENELITIAN

Metode penelitian yang digunakan pada penelitian ini adalah metode analisis deskriptif kuantitatif dengan memanfaatkan algoritma K-Means untuk menganalisis cluster dari penyebaran penyakit HIV/AIDS di provinsi Jawa Barat. Adapun data yang digunakan dalam penelitian ini sebanyak 108 baris dan terdapat 8 kolom yaitu id, kode provinsi, nama provinsi, kode kabupaten/kota, nama kabupaten/kota, jumlah kasus, satuan, dan tahun.



Gambar 1. Desain sistem yang diusulkan dalam proyek

Untuk dapat melakukan pengembangan penelitian, diperlukan beberapa tahapan yang harus dilakukan dengan tujuan mengolah sumber data menjadi data yang baik dan terstruktur, seperti:

3.1 Data Collecting

Pada tahap ini dilakukan proses membaca data dari file CSV ke dalam DataFrame, dan kemudian melakukan berbagai operasi analisis, manipulasi, dan visualisasi data menggunakan library Pandas, NumPy, dan Matplotlib.

3.2 Pra-process

Pra-process merupakan proses pembersihan data dari missing value atau noise yaitu data yang tidak relevan atau tidak konsisten. Pada tahap preprocessing ini, data yang diperoleh dari mengambil 3 kolom yang diperlukan, memisahkannya pertahun, dan menghapus kolom tertentu sesuai kebutuhan.

3.3 Eksplorasi Data Analisis

Dalam eksplorasi data analisis, langkah-langkah umumnya meliputi memeriksa dimensi data, tipe data, jumlah nilai null pada setiap kolom, dan memberikan ringkasan statistik dari dataframe.

3.4 Pemodelan data

Dalam pemodelan data, terdapat dua cara yang dapat digunakan. Pertama, menggunakan library scikit-learn. Kedua, tanpa menggunakan library scikit-learn. Pada cara pertama, digunakan satu set data dari tahun 2018. Sedangkan, pada cara kedua, digunakan dua set data dari tahun 2018 dan 2019. Dalam kedua cara tersebut, dilakukan visualisasi data yang akan diklasterisasi untuk membantu proses klasterisasi data tersebut.

3.5 Evaluasi performa

Pada tahap interpretasi atau evaluasi, dilakukan dengan menganalisa hasil eksperimen yang telah dilakukan.

BAB IV: HASIL DAN PEMBAHASAN

4.1 Data collecting

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt


#membuat data file ke DataFrame

df = pd.read_csv('/content/drive/MyDrive/Sains
Data/Semester 4/PJBL Data Mininh/data.csv')

df
```

Yang pertama, meng-import modul yang akan digunakan. antara lain pandas, numpy, dan matplotlib.pyplot. pada baris ini dilakukan pembacaan file dalam bentuk csv dan menampilkan file ini dalam bentuk data frame bernama df.

4.2 Pra-Process

```
# Menggunakan metode loc untuk mengambil 3 kolom yang
diperlukan

df1 = df.loc[:, ['nama_kabupaten_kota', 'tahun',
'jumlah_kasus']]

print(df1)
```

Output :

	nama_kabupaten_kota	tahun	jumlah_kasus
0	KABUPATEN BOGOR	2018	318
1	KABUPATEN SUKABUMI	2018	112
2	KABUPATEN CIANJUR	2018	124
3	KABUPATEN BANDUNG	2018	219
4	KABUPATEN GARUT	2018	107
..
103	KOTA BEKASI	2021	390
104	KOTA DEPOK	2021	199
105	KOTA CIMAHI	2021	342
106	KOTA TASIKMALAYA	2021	99
107	KOTA BANJAR	2021	40
[108 rows x 3 columns]			

Gambar 2. Output

Pada pra-process , Pertama-tama dilakukan pengambilan 3 kolom yang diperlukan menggunakan metode loc. kolom yang digunakan antara lain nama_kabupaten_kota, tahun, jumlah_kasus. tiga kolom ini di simpan pada variabel bernama df satu.

```
# memisahkan pertahun
df_2018 = df1[df1['tahun'] == 2018]
df_2019 = df1[df1['tahun'] == 2019]
df_2020 = df1[df1['tahun'] == 2020]
df_2021 = df1[df1['tahun'] == 2021]
```

Selanjutnya , data dipisahkan berdasarkan tahun. Hal ini dilakukan dengan mengambil kolom tahun pada df satu sesuai dengan tahun yang diperlukan. pada data yang di gunakan tahun yang di pilih antara lain 2018,2019,2020,dan 2021.

```
# menghapus kolom tahun dan merename kolom
'jumlah_kasus' dengan kasusnya per tahun

# 2018
df_2018 = df_2018.rename(columns={'jumlah_kasus':
'2018'})
```



```

df_2018 = df_2018.drop('tahun', axis=1)

# 2019
df_2019 = df_2019.rename(columns={'jumlah_kasus':
'2019'})
df_2019 = df_2019.drop('tahun', axis=1)

# 2020
df_2020 = df_2020.rename(columns={'jumlah_kasus':
'2020'})
df_2020 = df_2020.drop('tahun', axis=1)

# 2021
df_2021 = df_2021.rename(columns={'jumlah_kasus':
'2021'})
df_2021 = df_2021.drop('tahun', axis=1)

```

Pada kodingan ini, dilakukan penghapusan kolom tahun dan me-rename kolom jumlah_kasus dengan tahun kasusnya.

```

# menggabungkan kolom
df_merged = pd.merge(df_2018, df_2019,
on='nama_kabupaten_kota')

df_merged = pd.merge(df_merged, df_2020,
on='nama_kabupaten_kota')

df_merged = pd.merge(df_merged, df_2021,
on='nama_kabupaten_kota')

df_merged

```

Output :

	nama_kabupaten_kota	2018	2019	2020	2021
0	KABUPATEN BOGOR	318	475	417	430
1	KABUPATEN SUKABUMI	112	112	110	117
2	KABUPATEN CIANJUR	124	0	189	111
3	KABUPATEN BANDUNG	219	179	176	225
4	KABUPATEN GARUT	107	0	5	172
5	KABUPATEN TASIKMALAYA	40	77	95	68
6	KABUPATEN CIAMIS	66	95	62	58
7	KABUPATEN KUNINGAN	78	91	48	113
8	KABUPATEN CIREBON	278	257	251	232
9	KABUPATEN MAJALENGKA	92	97	87	123
10	KABUPATEN SUMEDANG	61	104	100	120
11	KABUPATEN INDRAMAYU	407	151	275	113
12	KABUPATEN SUBANG	230	339	247	222
13	KABUPATEN PURWAKARTA	194	197	234	130
14	KABUPATEN KARAWANG	92	255	315	244
15	KABUPATEN BEKASI	203	222	134	239
16	KABUPATEN BANDUNG BARAT	71	36	67	67
17	KABUPATEN PANGANDARAN	38	26	16	4
18	KOTA BOGOR	446	443	364	333
19	KOTA SUKABUMI	161	57	41	43
20	KOTA BANDUNG	1054	357	82	43
21	KOTA CIREBON	65	189	342	254
22	KOTA BEKASI	360	335	322	390
23	KOTA DEPOK	227	247	220	199
24	KOTA CIMAHI	104	44	361	342
25	KOTA TASIKMALAYA	121	105	130	99
26	KOTA BANJAR	7	47	68	40

Gambar 3. Output

Kodingan ini melakukan penggabungan empat Dataframe berbeda, yaitu `df_2018`, `df_2019`, `df_2020`, dan `df_2021`, berdasarkan kolom 'nama_kabupaten_kota'. Hasil penggabungan disimpan dalam variabel `df_merged`. Dengan melakukan penggabungan ini, dapat menggabungkan data dari tahun 2018,2019,2020 dan 2021 ke dalam satu Dataframe tunggal.

4.3 Eksplorasi Data Analisis

```
# memeriksa dimensi data  
  
import numpy as np  
  
np.shape(df_merged)
```

Output :



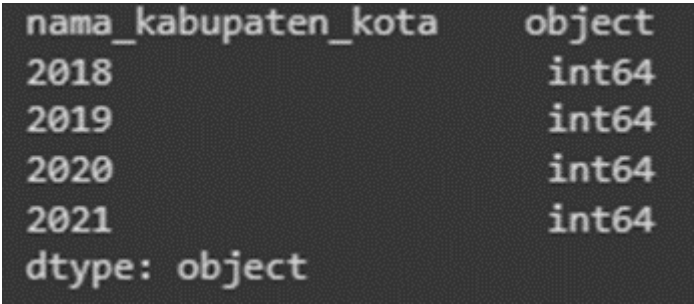
```
(27, 5)
```

Gambar 4. Output

disini menggunakan NumPy untuk memberikan informasi tentang jumlah baris dan kolom dalam DataFrame `df_merged`.

```
df_merged.dtypes
```

Output :



```
nama_kabupaten_kota    object  
2018                    int64  
2019                    int64  
2020                    int64  
2021                    int64  
dtype: object
```

Gambar 5. Output

Pada kodingan ini, `df_merged.dtypes` digunakan untuk menampilkan tipe data dari setiap kolom dalam DataFrame `df_merged`. Pada output, menampilkan tipe data dari masing-masing kolom dalam DataFrame. Kolom `nama_kabupaten_kota` memiliki tipe data `object`, sedangkan kolom 2018, 2019, 2020, dan 2021 memiliki tipe data `int64` atau integer 64-bit.

```
df_merged.isnull().sum()
```

Output :

```
nama_kabupaten_kota    0
2018                    0
2019                    0
2020                    0
2021                    0
dtype: int64
```

Gambar 6. Output

Kodingan tersebut digunakan untuk mengecek keberadaan nilai null atau kosong dalam setiap kolom DataFrame `df_merged`. pada outputnya menunjukkan jumlah nilai null dalam setiap kolom. Dalam kasus tersebut, tidak ada nilai null yang ditemukan dalam kolom-kolom `nama_kabupaten_kota`, 2018, 2019, 2020, dan 2021.

```
df_merged.describe()
```

Output :

	2018	2019	2020	2021
count	27.00000	27.000000	27.000000	27.000000
mean	195.37037	168.037037	176.222222	167.814815
std	207.39597	133.188554	121.702453	113.365317
min	7.00000	0.000000	5.000000	4.000000
25%	74.50000	67.000000	75.000000	83.500000
50%	121.00000	112.000000	134.000000	123.000000
75%	228.50000	251.000000	263.000000	235.500000
max	1054.00000	475.000000	417.000000	430.000000

Gambar 7. Output

Disini menggunakan metode `describe` pada DataFrame `df_merged` untuk menghasilkan statistik deskriptif dari kolom-kolom numerik. Pada output, menampilkan jumlah data non-null (`count`), rata-rata (`mean`), standar deviasi (`std`), nilai terkecil (`min`), kuartil (25%, 50%, 75%), dan nilai terbesar (`max`) dari setiap kolom numerik dalam DataFrame. Ini memberikan gambaran singkat tentang sebaran dan karakteristik statistik dari data dalam DataFrame `df_merged`.

4.4 Pemodelan Data

- Klasterisasi tahun 2018(X) - 2019(Y)

```
# mengambil 2 data yang akan diklasterisasi  
data = df_merged.loc[:, ['2018', '2019']]  
data.head(2)
```

Code di atas digunakan untuk mengambil dua kolom 2018 dan 2019 dari dataframe df_merged dan menyimpannya dalam dataframe data.

```
# convert to numpy array  
X = data.values  
X
```

Code di atas digunakan untuk mengonversi dataframe data menjadi array numpy menggunakan metode values.

Output :

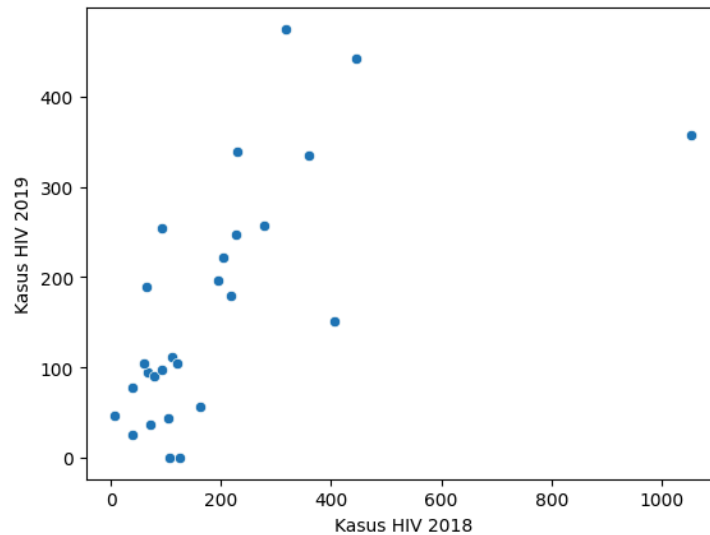
```
array([[ 318,  475],  
       [ 112,  112],  
       [ 124,    0],  
       [ 219, 179],  
       [ 107,    0],  
       [  40,  77],  
       [  66,  95],  
       [  78,  91],  
       [ 278, 257],  
       [  92,  97],  
       [  61, 104],  
       [ 407, 151],  
       [ 230, 339],  
       [ 194, 197],  
       [  92, 255],  
       [ 203, 222],  
       [  71,  36],  
       [  38,  26],  
       [ 446, 443],  
       [ 161,  57],  
       [1054, 357],  
       [  65, 189],  
       [ 360, 335],  
       [ 227, 247],  
       [ 104,  44],  
       [ 121, 105],  
       [   7,  47]])
```

Gambar 8. Output

```
# visualisasi data dengan library seaborn  
sns.scatterplot(x=X[:,0], y=X[:, 1])
```

```
plt.xlabel('Kasus HIV 2018')
plt.ylabel('Kasus HIV 2019')
plt.show()
```

Code di atas digunakan untuk melakukan visualisasi data pada tahun 2018 dan 2019
Output:



Gambar 9. Grafik

```
# kalkulasi WCSS
def calculate_cost(X, centroids, cluster):
    sum = 0
    for i, val in enumerate(X):
        sum += np.sqrt((centroids[int(cluster[i])],
0]-val[0])**2 +(centroids[int(cluster[i])],
1]-val[1])**2)
    return sum
```

Code di atas digunakan untuk menghitung Within-Cluster Sum of Squares (WCSS) atau jumlah kuadrat dalam klaster.

```
# implementasi kmeans
```

```

def kmeans(X, k):
    diff = 1
    cluster = np.zeros(X.shape[0])
    centroids = data.sample(n=k).values
    while diff:
        # for each observation
        for i, row in enumerate(X):
            mn_dist = float('inf')
            # dist of the point from all centroids
            for idx, centroid in enumerate(centroids):
                d = np.sqrt((centroid[0]-row[0])**2 +
                    (centroid[1]-row[1])**2)
                # store closest centroid
                if mn_dist > d:
                    mn_dist = d
                    cluster[i] = idx
            new_centroids =
pd.DataFrame(X).groupby(by=cluster).mean().values
            # if centroids are same then leave
            if np.count_nonzero(centroids-new_centroids) ==
0:
                diff = 0
            else:
                centroids = new_centroids
    return centroids, cluster

```

code di atas digunakan untuk mengimplementasikan algoritma K-Means untuk melakukan klasterisasi data.

```
# mencari K value dengan metode Elbow

cost_list = []

for k in range(1, 10):

    centroids, cluster = kmeans(X, k)

    # WCSS (Within cluster sum of square)

    cost = calculate_cost(X, centroids, cluster)

    cost_list.append(cost)
```

Code di atas digunakan untuk mencari nilai optimal K (jumlah klaster) menggunakan metode Elbow. Dengan menggunakan metode Elbow, kita dapat mencari titik di grafik di mana penurunan WCSS mulai melambat secara signifikan (menyerupai siku). Titik ini menunjukkan jumlah cluster yang optimal, di mana penambahan klaster lebih lanjut memberikan manfaat yang lebih kecil dalam mengurangi WCSS.

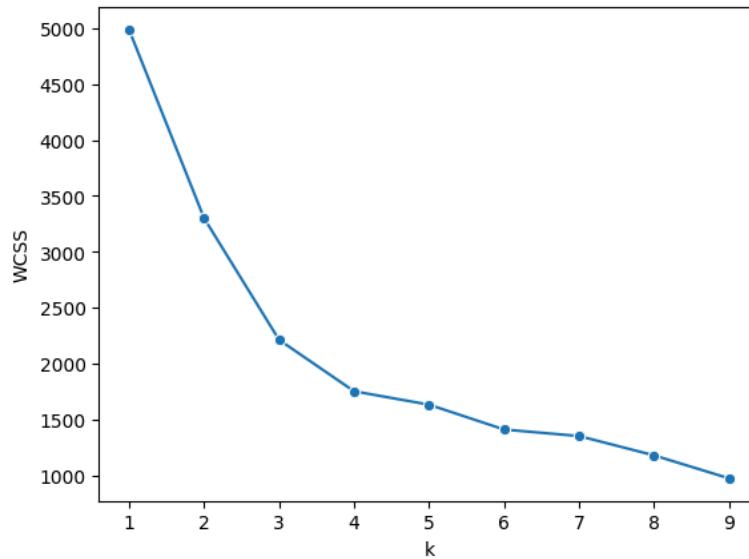
```
sns.lineplot(x=range(1,10), y=cost_list, marker='o')

plt.xlabel('k')

plt.ylabel('WCSS')

plt.show()
```

Code di atas digunakan untuk melakukan visualisasi grafik Elbow, yang dapat membantu menentukan nilai optimal K (jumlah klaster) dalam metode klasterisasi. Output:



Gambar 10. Grafik

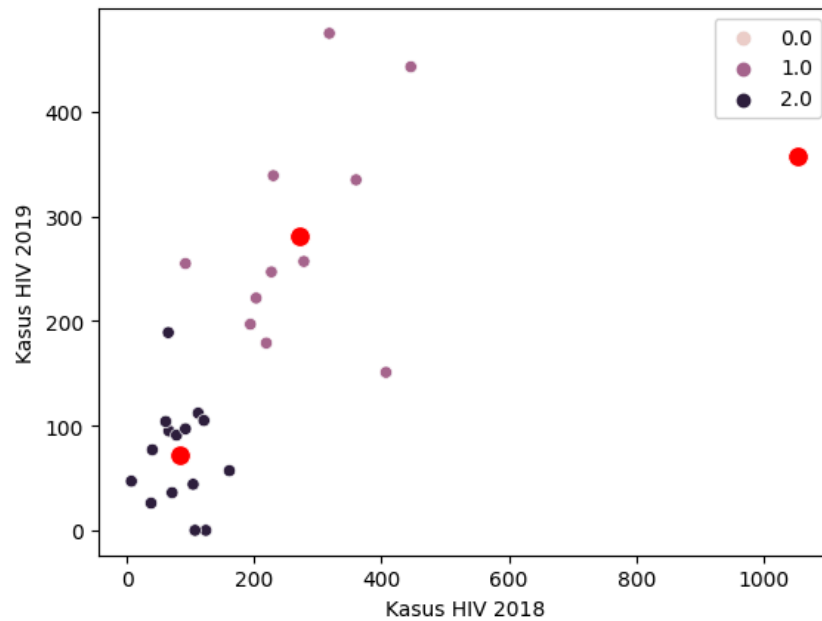
```
# membuat klaster
k = 3
centroids, cluster = kmeans(X, k)
```

code di atas digunakan untuk membuat cluster menggunakan algoritma K-Means dengan jumlah klaster (k) 3

```
# Visualisasi Klaster
sns.scatterplot(x=X[:,0], y=X[:, 1], hue=cluster)
sns.scatterplot(x=centroids[:,0], y=centroids[:, 1],
s=100, color='red')
plt.xlabel('Kasus HIV 2018')
plt.ylabel('Kasus HIV 2019')
plt.show()
```

code di atas digunakan untuk melakukan visualisasi klaster setelah proses klasterisasi menggunakan algoritma K-Means.

Output :



Gambar 11. Grafik

```
# hasil klasterisasi pada semua tahun
df_merged['cluster'] = cluster
df_merged
```

Code di atas digunakan untuk menambahkan kolom baru bernama 'cluster' ke dalam DataFrame df_merged yang berisi label klaster untuk setiap data (0.0 = HIV rendah, 1.0 = HIV sedang, 2.0 = HIV tinggi).

Output :

	nama_kabupaten_kota	2018	2019	2020	2021	cluster
0	KABUPATEN BOGOR	318	475	417	430	1.0
1	KABUPATEN SUKABUMI	112	112	110	117	2.0
2	KABUPATEN CIANJUR	124	0	189	111	2.0
3	KABUPATEN BANDUNG	219	179	176	225	1.0
4	KABUPATEN GARUT	107	0	5	172	2.0
5	KABUPATEN TASIKMALAYA	40	77	95	68	2.0
6	KABUPATEN CIAMIS	66	95	62	58	2.0
7	KABUPATEN KUNINGAN	78	91	48	113	2.0
8	KABUPATEN CIREBON	278	257	251	232	1.0
9	KABUPATEN MAJALENGKA	92	97	87	123	2.0
10	KABUPATEN SUMEDANG	61	104	100	120	2.0
11	KABUPATEN INDRAMAYU	407	151	275	113	1.0
12	KABUPATEN SUBANG	230	339	247	222	1.0
13	KABUPATEN PURWAKARTA	194	197	234	130	1.0
14	KABUPATEN KARAWANG	92	255	315	244	1.0

Gambar 12. Output

```
centroid
```

Code di atas digunakan untuk mengakses dan menampilkan pusat kluster yang dihasilkan dari proses klasterisasi menggunakan algoritma K-Means.

output :

```
array([[1054.      , 357.      ],
       [ 270.36363636, 281.81818182],
       [ 83.13333333, 72.      ]])
```

Gambar 13. Output

4.5 Evaluasi Performa

- indeks Davies-Bouldin

```
def davies_bouldin_index(X, clusters):
    num_clusters = len(np.unique(clusters))
    cluster_centers = []
    for k in range(num_clusters):
        cluster_points = X[clusters == k]
        cluster_center = np.mean(cluster_points, axis=0)
        cluster_centers.append(cluster_center)
    db = 0
    for i in range(num_clusters):
        for j in range(num_clusters):
            if i != j:
                s_i = np.mean([np.linalg.norm(X[p] -
                    cluster_centers[i]) for p in range(len(X)) if
                    clusters[p] == i])
                s_j = np.mean([np.linalg.norm(X[q] -
                    cluster_centers[j]) for q in range(len(X)) if
                    clusters[q] == j])
```

```

        m_ij =
np.linalg.norm(cluster_centers[i] -
cluster_centers[j])

        db += (s_i + s_j) / m_ij

    return db / num_clusters

# Contoh penggunaan

davies_bouldin_score = davies_bouldin_index(X,
cluster)

print("Davies-Bouldin Index:", davies_bouldin_score)

```

Output:

```
Davies-Bouldin Index: 0.5719518067907118
```

Gambar 14. Output

Code di atas adalah implementasi dari fungsi `davies_bouldin_index`. Fungsi ini menghitung indeks Davies-Bouldin untuk sebuah set data yang dikelompokkan menjadi beberapa kluster. Indeks Davies-Bouldin digunakan untuk mengevaluasi kualitas klustering pada data.

- koefisien Silhouette

```

def silhouette_coefficient(X, clusters):

    num_samples = len(X)

    s = 0

    for i in range(num_samples):

        cluster_i = clusters[i]

        cluster_i_samples = X[clusters == cluster_i]

        a = np.mean([np.linalg.norm(X[i] - sample) for sample in
cluster_i_samples if sample is not X[i]])

```

```

b_values = []

unique_clusters = np.unique(clusters)

for k in unique_clusters:

    if k != cluster_i:

        cluster_k_samples = X[clusters == k]

        b = np.mean([np.linalg.norm(X[i] - sample) for
sample in cluster_k_samples])

        b_values.append(b)

if len(b_values) > 0:

    b = np.min(b_values)

    s += (b - a) / max(a, b)

if num_samples > 0:
return s / num_samples
else:
return 0.0

# Example usage
silhouette_score = silhouette_coefficient(X, cluster)
print("Silhouette Coefficient:", silhouette_score)

```

output::

```
Silhouette Coefficient: 0.5931304129939626
```

Gambar 15. Output


Code di atas adalah implementasi dari fungsi `silhouette_coefficient`. Fungsi ini digunakan untuk menghitung koefisien Silhouette, yang merupakan metrik evaluasi kualitas klastering pada data.

- **Klasterisasi tahun 2020(X) - 2021(Y)**

```
# mengambil 2 data yang akan diklasterisasi  
data = df_merged.loc[:, ['2020', '2021']]  
data.head(2)
```

code di atas digunakan untuk mengambil dua kolom 2020 dan 2021 dari dataframe `df_merged` dan menyimpannya dalam dataframe `data`.

Output



	2020	2021
0	417	430
1	110	117

Gambar 16. Output

```
# convert to numpy array

X = data.values

X
```

code di atas digunakan untuk mengonversi dataframe data menjadi array numpy menggunakan metode values

Output

```
array([[417, 430],
       [110, 117],
       [189, 111],
       [176, 225],
       [  5, 172],
       [ 95,  68],
       [ 62,  58],
       [ 48, 113],
       [251, 232],
       [ 87, 123],
       [100, 120],
       [275, 113],
       [247, 222],
       [234, 130],
       [315, 244],
       [134, 239],
       [ 67,  67],
       [ 16,   4],
       [364, 333],
       [ 41,  43],
       [ 82,  43],
       [342, 254],
       [322, 390],
       [220, 199],
       [361, 342],
       [130,  99],
       [ 68,  40]])
```

Gambar 17. Output

```
# mencari K value dengan metode Elbow

cost_list = []

for k in range(1, 10):

    centroids, cluster = kmeans(X, k)

    # WCSS (Within cluster sum of square)

    cost = calculate_cost(X, centroids, cluster)

    cost_list.append(cost)
```

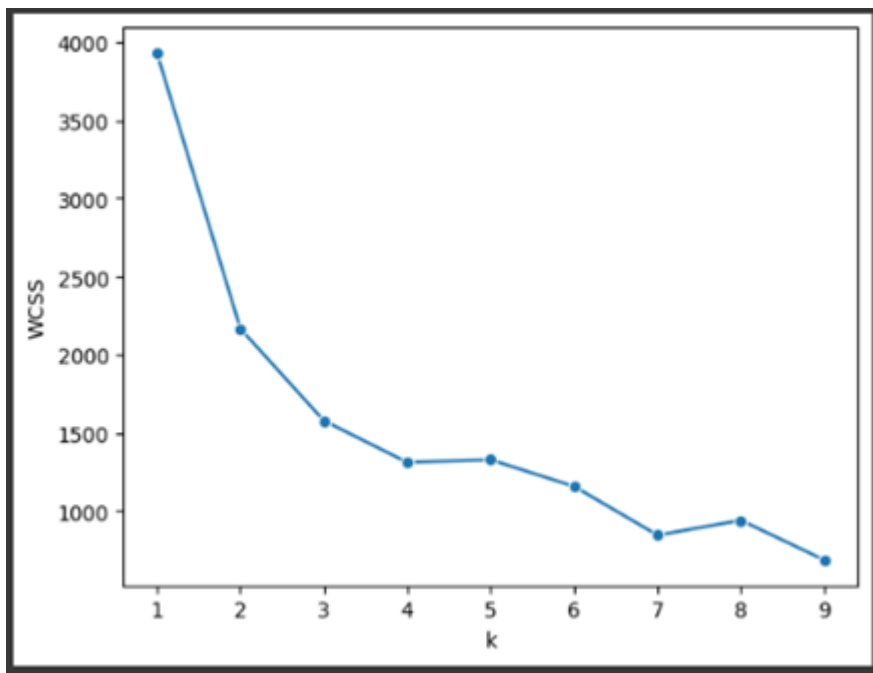
code di atas digunakan untuk mencari nilai optimal K (jumlah klaster) menggunakan metode Elbow. Dengan menggunakan metode Elbow, kita dapat mencari titik di grafik di mana

penurunan WCSS mulai melambat secara signifikan (menyerupai siku). Titik ini menunjukkan jumlah cluster yang optimal, di mana penambahan klaster lebih lanjut memberikan manfaat yang lebih kecil dalam mengurangi WCSS

```
sns.lineplot(x=range(1,10), y=cost_list, marker='o')  
  
plt.xlabel('k')  
  
plt.ylabel('WCSS')  
  
plt.show()
```

code di atas digunakan untuk melakukan visualisasi grafik Elbow, yang dapat membantu menentukan nilai optimal K (jumlah klaster) dalam metode klasterisasi

Output:



Gambar 18. Grafik

```
# membuat klaster  
  
k = 2  
  
centroids, cluster = kmeans(X, k)
```


code di atas digunakan untuk membuat cluster menggunakan algoritma K-Means dengan jumlah kluster (k) 3

```
# Visualisasi Klaster

sns.scatterplot(x=X[:,0], y=X[:, 1], hue=cluster)

sns.scatterplot(x=centroids[:,0], y=centroids[:, 1], s=100,
color='red')

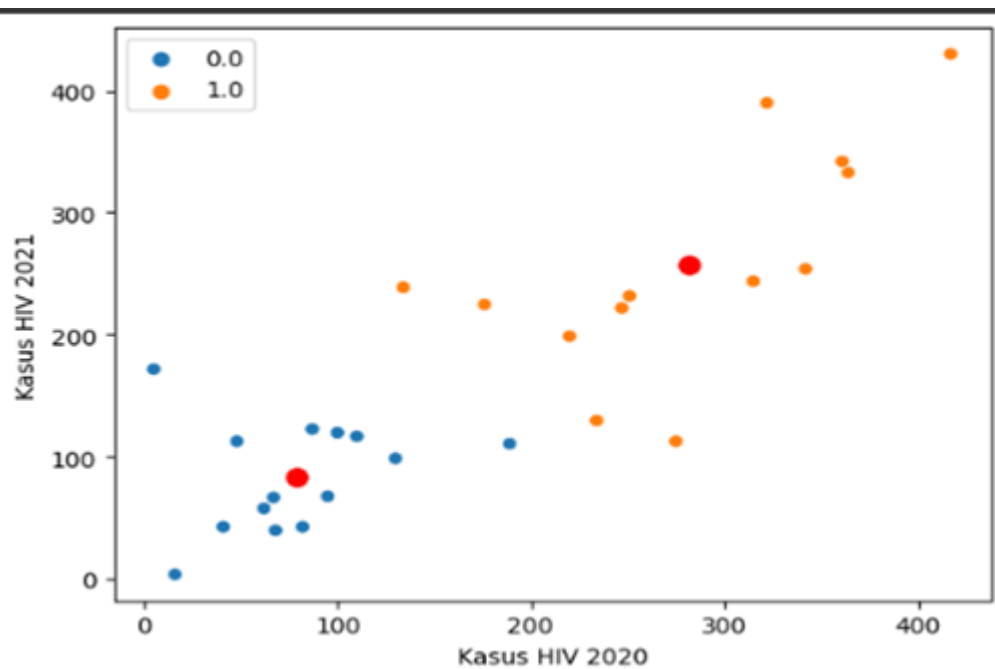
plt.xlabel('Kasus HIV 2020')

plt.ylabel('Kasus HIV 2021')

plt.show()
```

code di atas digunakan untuk melakukan visualisasi kluster setelah proses klusterisasi menggunakan algoritma K-Means.

Output



Gambar 19. Grafik

```
# hasil klasterisasi pada tahun 2020
# 0.0 = HIV rendah, 1.0 = HIV tinggi
df_merged['cluster'] = cluster
df_merged
```

code di atas digunakan untuk menambahkan kolom baru bernama 'cluster' ke dalam DataFrame df_merged yang berisi label klaster untuk setiap data (0.0 = HIV rendah, 1.0 = HIV sedang, 2.0 = HIV tinggi)

Output :

	nama_kabupaten_kota	2018	2019	2020	2021	cluster
0	KABUPATEN BOGOR	318	475	417	430	1.0
1	KABUPATEN SUKABUMI	112	112	110	117	0.0
2	KABUPATEN CIANJUR	124	0	189	111	0.0
3	KABUPATEN BANDUNG	219	179	176	225	1.0
4	KABUPATEN GARUT	107	0	5	172	0.0
6	KABUPATEN TASIKMALAYA	40	77	95	68	0.0
8	KABUPATEN CIAMIS	66	95	62	58	0.0
7	KABUPATEN KUNINGAN	78	91	48	113	0.0
8	KABUPATEN CIREBON	278	257	251	232	1.0
9	KABUPATEN MAJALENGKA	92	97	87	123	0.0
10	KABUPATEN SUMEDANG	61	104	100	120	0.0
11	KABUPATEN INDRAMAYU	407	151	275	113	1.0
12	KABUPATEN SUBANG	230	339	247	222	1.0
13	KABUPATEN PURWAKARTA	194	197	234	130	1.0
14	KABUPATEN KARAWANG	92	255	315	244	1.0
16	KABUPATEN BEKASI	203	222	134	239	1.0
18	KABUPATEN BANDUNG BARAT	71	36	67	67	0.0
17	KABUPATEN PANGANDARAN	38	26	16	4	0.0
18	KOTA BOGOR	446	443	364	333	1.0
19	KOTA SUKABUMI	161	57	41	43	0.0
20	KOTA BANDUNG	1054	357	82	43	0.0
21	KOTA CIREBON	65	189	342	254	1.0
22	KOTA BEKASI	360	335	322	390	1.0
23	KOTA DEPOK	227	247	220	199	1.0
24	KOTA CIMAH	104	44	361	342	1.0
26	KOTA TASIKMALAYA	121	105	130	99	0.0
28	KOTA BANJAR	7	47	68	40	0.0

Gambar 20. Output

```
centroids
```

code di atas digunakan untuk mengakses dan menampilkan pusat kluster yang dihasilkan dari proses klasterisasi menggunakan algoritma K-Means

output

```
array([[ 78.57142857,  84.14285714],  
       [281.38461538, 257.92307692]])
```

Gambar 21. Output centroid

```

def davies_bouldin_index(X, clusters):
    num_clusters = len(np.unique(clusters))
    cluster_centers = []
    for k in range(num_clusters):
        cluster_points = X[clusters == k]
        cluster_center = np.mean(cluster_points, axis=0)
        cluster_centers.append(cluster_center)

    db = 0

    for i in range(num_clusters):
        for j in range(num_clusters):
            if i != j:
                s_i = np.mean([np.linalg.norm(X[p] -
cluster_centers[i]) for p in range(len(X)) if clusters[p]
== i])
                s_j = np.mean([np.linalg.norm(X[q] -
cluster_centers[j]) for q in range(len(X)) if clusters[q]
== j])
                m_ij = np.linalg.norm(cluster_centers[i] -
cluster_centers[j])
                db += (s_i + s_j) / m_ij

    return db / num_clusters

# Contoh penggunaan
davies_bouldin_score = davies_bouldin_index(X, cluster)
print("Davies-Bouldin Index:", davies_bouldin_score)

```

Code di atas adalah implementasi dari fungsi `davies_bouldin_index`. Fungsi ini menghitung indeks Davies-Bouldin untuk sebuah set data yang dikelompokkan menjadi beberapa kluster. Indeks Davies-Bouldin digunakan untuk mengevaluasi kualitas klustering pada data.

Output

Davies-Bouldin Index: 0.6081815042659908

Gambar 22. Output

```
def silhouette_coefficient(X, clusters):  
    num_samples = len(X)  
    s = 0  
    for i in range(num_samples):  
        cluster_i = clusters[i]  
        cluster_i_samples = X[clusters == cluster_i]  
        a = np.mean([np.linalg.norm(X[i] - sample) for  
sample in cluster_i_samples if sample is not X[i]])  
  
        b_values = []  
        unique_clusters = np.unique(clusters)  
        for k in unique_clusters:  
            if k != cluster_i:  
                cluster_k_samples = X[clusters == k]  
                b = np.mean([np.linalg.norm(X[i] - sample)  
for sample in cluster_k_samples])  
                b_values.append(b)  
  
        if len(b_values) > 0:  
            b = np.min(b_values)  
            s += (b - a) / max(a, b)
```

```

    if num_samples > 0:
        return s / num_samples
    else:
        return 0.0

# Example usage
silhouette_score = silhouette_coefficient(X, cluster)
print("Silhouette Coefficient:", silhouette_score)

```

Code di atas adalah implementasi dari fungsi `silhouette_coefficient`. Fungsi ini digunakan untuk menghitung koefisien Silhouette, yang merupakan metrik evaluasi kualitas klastering pada data

Output :

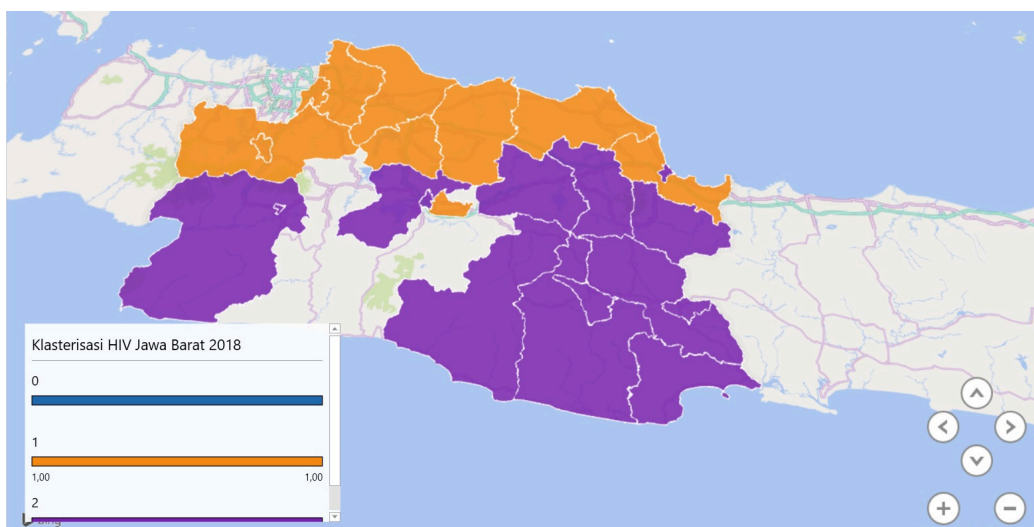
```

Silhouette Coefficient: 0.5781416778253102

```

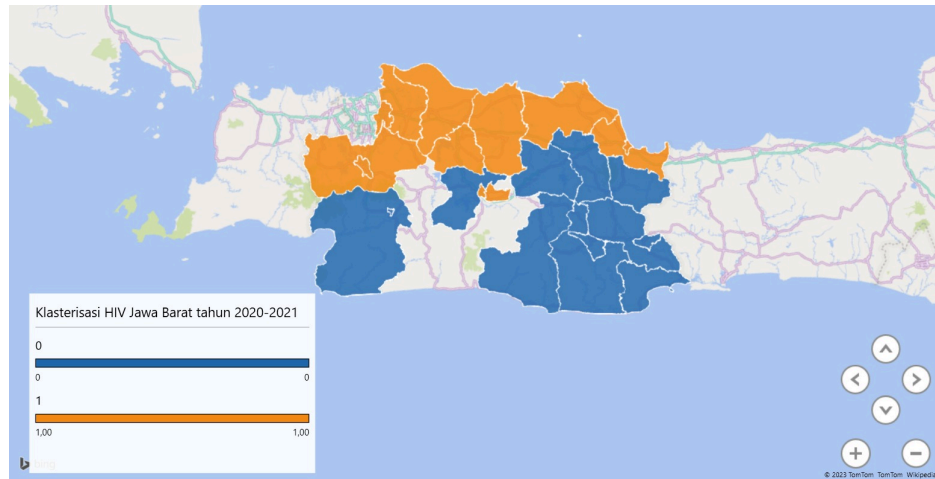
Gambar 23. Output

Visualisasi hasil klasterisasi tahun 2018-2019 pada 3D maps menggunakan Excel



Gambar 24. Peta Persebaran

Visualisasi hasil klasterisasi tahun 2020-2021 pada 3D maps menggunakan Excel



Gambar 25. Peta Persebaran

BAB V: KESIMPULAN

Pada dataset kami yaitu jumlah kasus HIV Jawa barat pada tahun 2018-2021 dapat dilakukan klasterisasi dengan metode K-Means. Pada tahun 2018-2019 kami membagi 3 klaster dengan didapatkan hasil klaster 0 yaitu daerah dengan kasus HIV paling tinggi, klaster 1 yaitu dengan kasus HIV sedang dan klaster 2 dengan kasus HIV paling rendah. Pada tahun 2020-2021 kami membagi 2 klaster dengan didapatkan hasil klaster 0 yaitu daerah dengan kasus HIV rendah dan klaster 1 yaitu daerah dengan kasus HIV tinggi.

Pada Evaluasi performa pada klasterisasi tahun 2018-2019 didapatkan hasil evaluasi performa dengan davies bouldin index sebesar 0,572 dan kami juga menguji hasil evaluasi performa menggunakan silhouette coefficient didapatkan hasil sebesar 0.602. Sedangkan pada klasterisasi tahun 2020-2021 didapatkan hasil evaluasi performa dengan davies bouldin index sebesar 0,608 dan silhouette coefficient didapatkan hasil sebesar 0.578. Dimana jika nilai akurasi dari davies bouldin index dan silhouette coefficient mendekati 1, maka semakin akurat juga model yang dibuat.

DAFTAR PUSTAKA

- Sinaga, R. I. L., Saputra, W., & Qurniawan, H. (2021). Pengelompokan Jumlah Kasus Penyakit Aids Berdasarkan Provinsi Menggunakan Metode K-Means. *Kesatria : Jurnal Penerapan Sistem Informasi (Komputer Dan Manajemen)*, 2(2), Article 2. <https://doi.org/10.30645/kesatria.v2i2.64>
- Rosida, W., & Wijaya, Y. A. (2023). Klasterisasi Penyakit HIV/AIDS di Jawa Barat Menggunakan Algoritma K-Means Clustering. *Blend Sains Jurnal Teknik*, 1(4), Article 4. <https://doi.org/10.56211/blendsains.v1i4.235>
- Sari, T. P., Hananto, A. L., Novalia, E., Tukino, T., & Hilabi, S. S. (2023). Implementasi Algoritma K-Means dalam Analisis Klasterisasi Penyebaran Penyakit Hiv/Aids. *Infotek: Jurnal Informatika Dan Teknologi*, 6(1), Article 1. <https://doi.org/10.29408/jit.v6i1.7423>
- Noor, H., Dharmawati, A., & Qur'ana, T. W. (2021). PENERAPAN ALGORITMA K-MEANS CLUSTERING ANALYSIS PADA KASUS PENDERITA HIV/AIDS (STUDI KASUS KABUPATEN BANJAR). *Technologia : Jurnal Ilmiah*, 12(2), Article 2. <https://doi.org/10.31602/tji.v12i2.4573>
- HIV dan AIDS - Faktor Risiko, Gejala, dan Penanganannya*. (n.d.). HIV dan AIDS - Faktor Risiko, Gejala, dan Penanganannya. Retrieved May 31, 2023, from <https://www.siloamhospitals.com/informasi-siloam/artikel/apa-itu-hiv>
- Lesmana, A. A., Purwanto, Y., & Dinimaharawati, A. (2019). Implementasi Algoritma K-means Untuk Clustering Penyakit Hiv/aids Di Indonesia. *eProceedings of Engineering*, 6(2), Article 2. <https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/10578>
- Wasnik, A. (2020, December 31). *K-Means Clustering From Scratch in Python [Algorithm Explained]*—AskPython. <https://www.askpython.com/python/examples/k-means-clustering-from-scratch>
- How to program the kmeans algorithm in Python from scratch*—Ander Fernández. (n.d.). Retrieved May 31, 2023, from <https://anderfernandez.com/en/blog/kmeans-algorithm-python/>

LAMPIRAN