

TUGAS CASE BASED 2
MACHINE LEARNING



Nama:

Hilman Taris Muttaqin 1301204208

Anda mengerjakan tugas ini dengan cara yang tidak melanggar aturan perkuliahan dan kode etik akademisi.

Kode Dosen:

DDR

UNIVERSITAS TELKOM
TAHUN AKADEMIK 2022/2023

Bagian 1

Ikhtisar data

Data yang digunakan adalah data yang terdapat pada Kaggle tentang perkembangan suatu negara

<https://www.kaggle.com/datasets/rohan0301/unsupervised-learning-on-country-data>

Tujuan dari data ini adalah untuk mengkategorikan negara menggunakan faktor sosial-ekonomi dan kesehatan yang menentukan pembangunan negara secara keseluruhan. Terdapat 9 kolom pada data ini, yaitu:

1. country: nama dari negara
2. child_mort: kematian anak umur dibawah 5 tahun per 1000 kelahiran
3. exports: jumlah ekspor barang dan servis perkapita. Diberikan sebagai persentase dari GDP perkapita
4. health: jumlah pengeluaran untuk biaya kesehatan perkapita. Diberikan sebagai persentase GDP perkapita
5. imports: jumlah import barang dan servis perkapita. Diberikan sebagai persentase GDP perkapita
6. Income: pendapatan bersih perorang
7. Inflation: pengukuran dari pertambahan jumlah GDP pertahun
8. life_expec: rata-rata jumlah tahun dari anak yang belum lahir akan hidup jika pola mortalitas tetap sama
9. total_fer: banyak anak yang akan lahir untuk setiap wanita jika age-fertility sekarang masih sama
10. gdpp: GDP perkapita dihitung dari total GDP dan dibagi dengan banyaknya populasi

Preview Data

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	Afghanistan	90.2	10.0	7.58	44.9	1610	9.44	56.2	5.82	553
1	Albania	16.6	28.0	6.55	48.6	9930	4.49	76.3	1.65	4090
2	Algeria	27.3	38.4	4.17	31.4	12900	16.10	76.5	2.89	4460
3	Angola	119.0	62.3	2.85	42.9	5900	22.40	60.1	6.16	3530
4	Antigua and Barbuda	10.3	45.5	6.03	58.9	19100	1.44	76.8	2.13	12200
...
162	Vanuatu	29.2	46.6	5.25	52.7	2950	2.62	63.0	3.50	2970
163	Venezuela	17.1	28.5	4.91	17.6	16500	45.90	75.4	2.47	13500
164	Vietnam	23.3	72.0	6.84	80.2	4490	12.10	73.1	1.95	1310
165	Yemen	56.3	30.0	5.18	34.4	4480	23.60	67.5	4.67	1310
166	Zambia	83.1	37.0	5.89	30.9	3280	14.00	52.0	5.40	1460

167 rows × 10 columns

Informasi Umum

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 10 columns):
#   Column      Non-Null Count  Dtype
---  -
0   country     167 non-null   object
1   child_mort  167 non-null   float64
2   exports     167 non-null   float64
3   health      167 non-null   float64
4   imports     167 non-null   float64
5   income      167 non-null   int64
6   inflation   167 non-null   float64
7   life_expec  167 non-null   float64
8   total_fer   167 non-null   float64
9   gdpp        167 non-null   int64
dtypes: float64(7), int64(2), object(1)
memory usage: 13.2+ KB
```

Jika dilihat dari data ini, terdapat 167 record dan memiliki 1 tipe data deskriptif yaitu yang terdapat pada kolom country. Sehingga untuk kolom ini, akan di drop saat proses perhitungan. Tidak terdapat data yang kosong atau null.

General Descriptive Statistics

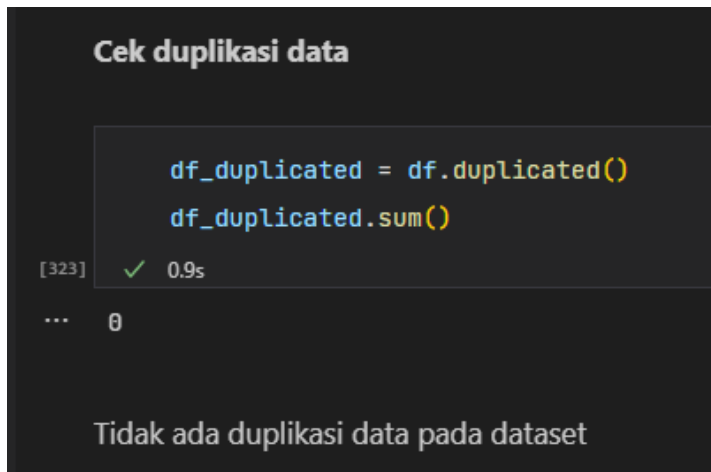
	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
count	167.000000	167.000000	167.000000	167.000000	167.000000	167.000000	167.000000	167.000000	167.000000
mean	38.270060	41.108976	6.815689	46.890215	17144.688623	7.781832	70.555689	2.947964	12964.155689
std	40.328931	27.412010	2.746837	24.209589	19278.067698	10.570704	8.893172	1.513848	18328.704809
min	2.600000	0.109000	1.810000	0.065900	609.000000	-4.210000	32.100000	1.150000	231.000000
25%	8.250000	23.800000	4.920000	30.200000	3355.000000	1.810000	65.300000	1.795000	1330.000000
50%	19.300000	35.000000	6.320000	43.300000	9960.000000	5.390000	73.100000	2.410000	4660.000000
75%	62.100000	51.350000	8.600000	58.750000	22800.000000	10.750000	76.800000	3.880000	14050.000000
max	208.000000	200.000000	17.900000	174.000000	125000.000000	104.000000	82.800000	7.490000	105000.000000

Bagian 2

Ringkasan Pra-Pemrosesan Data

Pra-Pemrosesan data merupakan proses menyiapkan data agar nantinya proses perhitungan atau learning dapat berjalan optimal dan menghasilkan hasil yang memuaskan. Terdapat beberapa proses yang diimplementasikan, yaitu:

1. Pengecekan duplikasi data



The screenshot shows a Jupyter Notebook interface with a dark theme. At the top, the cell title is "Cek duplikasi data". The code cell contains two lines of Python code: `df_duplicated = df.duplicated()` and `df_duplicated.sum()`. Below the code, the output is displayed: `[323] ✓ 0.9s` on the first line, and `... 0` on the second line. At the bottom of the cell, a text message reads "Tidak ada duplikasi data pada dataset".

```
Cek duplikasi data
```

```
df_duplicated = df.duplicated()
df_duplicated.sum()
```

```
[323] ✓ 0.9s
... 0
```

Tidak ada duplikasi data pada dataset

2. Pengecekan data kosong

```
Data Kosong  
Cek apakah terdapat data kosong atau tidak  
  
df.isna().sum()  
[324] ✓ 0.8s  
... country      0  
    child_mort    0  
    exports       0  
    health        0  
    imports       0  
    income        0  
    inflation     0  
    life_expec    0  
    total_fer     0  
    gdpp          0  
    dtype: int64
```

3. Scaling

```
Min Max Scaling  
  
def min_max_scaling(df) :  
    return (df.iloc[:,1:10] - df.iloc[:,1:10].min()) / (df.iloc[:,1:10].max() - df.iloc[:,1:10].min())  
[325] ✓ 0.6s  
  
df_scaling = min_max_scaling(df)  
df_scaling  
[326] ✓ 0.1s  
...  
    child_mort  exports  health  imports  income  inflation  life_expec  total_fer  gdpp  
0      0.426485  0.049482  0.358608  0.257765  0.008047  0.126144  0.475345  0.736593  0.003073  
1      0.068160  0.139531  0.294593  0.279037  0.074933  0.080399  0.871795  0.078864  0.036833  
2      0.120253  0.191559  0.146675  0.180149  0.098809  0.187691  0.875740  0.274448  0.040365  
3      0.566699  0.311125  0.064636  0.246266  0.042535  0.245911  0.552268  0.790221  0.031488  
4      0.037488  0.227079  0.262275  0.338255  0.148652  0.052213  0.881657  0.154574  0.114242  
...  
162    0.129503  0.232582  0.213797  0.302609  0.018820  0.063118  0.609467  0.370662  0.026143  
163    0.070594  0.142032  0.192666  0.100809  0.127750  0.463081  0.854043  0.208202  0.126650  
164    0.100779  0.359651  0.312617  0.460715  0.031200  0.150725  0.808679  0.126183  0.010299  
165    0.261441  0.149536  0.209447  0.197397  0.031120  0.257000  0.698225  0.555205  0.010299  
166    0.391918  0.184556  0.253574  0.177275  0.021473  0.168284  0.392505  0.670347  0.011731  
  
167 rows x 9 columns
```

Scaling berfungsi untuk mempercepat proses perhitungan, karena data aslinya memiliki rentang angka yang sangat besar. Scaling yang digunakan adalah min max scaling dengan rentang dari 0 hingga 1.

Kesimpulannya data yang digunakan termasuk data yang lumayan rapih

Bagian 3

Implementasi Model

Model yang digunakan untuk proses clustering pada tugas ini adalah metode kMeans. kMeans terdiri dari parameter k untuk menentukan jumlah cluster yang akan dibuat pada data yang ada. Berikut adalah rangkaian prosesnya:

1. Generate random centroid

Centroid digenerate sebanyak fitur * k. Setiap fitur atau kolom memiliki k centroid. Centroid diambil dari sample atau 1 record acak pada dataset.

2. Iterasi hingga centroid tidak berubah

Dalam Iterasi kMeans, terdapat langkah langkah yang dilakukan, yaitu:

- a. Labeling

Labeling dilakukan untuk melabeli apakah data termasuk ke label a, b, c, dst. Banyaknya label ditentukan dari nilai k. Label dihitung berdasarkan centroid pertama atau initial centroid.

$$D(p, c)_n = \sqrt{\sum_{i=0}^n (p_i - c_i)^2}$$

```
def labeling(df, centroid):  
    distance = centroid.apply(lambda x : np.sqrt(((df - x) ** 2).sum(axis = 1))))  
    return distance.idxmin(axis = 1)
```

[338] ✓ 0.9s

- b. Centroid

Pada tahap ini centroid baru dihitung. Perhitungan dilakukan dengan cara hitung masing-masing centroid dengan data yang ada dan ambil jarak terdekatnya (bisa menggunakan rumus Euclidean distance) yang pada akhirnya data dengan jarak terdekat itu akan menentukan kelasnya atau labeling. Centroid baru akan terbentuk dari

perhitungan jumlah kelas dibagi dengan banyaknya jumlah kelas atau rata-ratanya.

```
def new_centroid(df, label):  
    return df.groupby(label).apply(lambda x : np.exp(np.log(x).mean())).T
```

[331] ✓ 0.1s

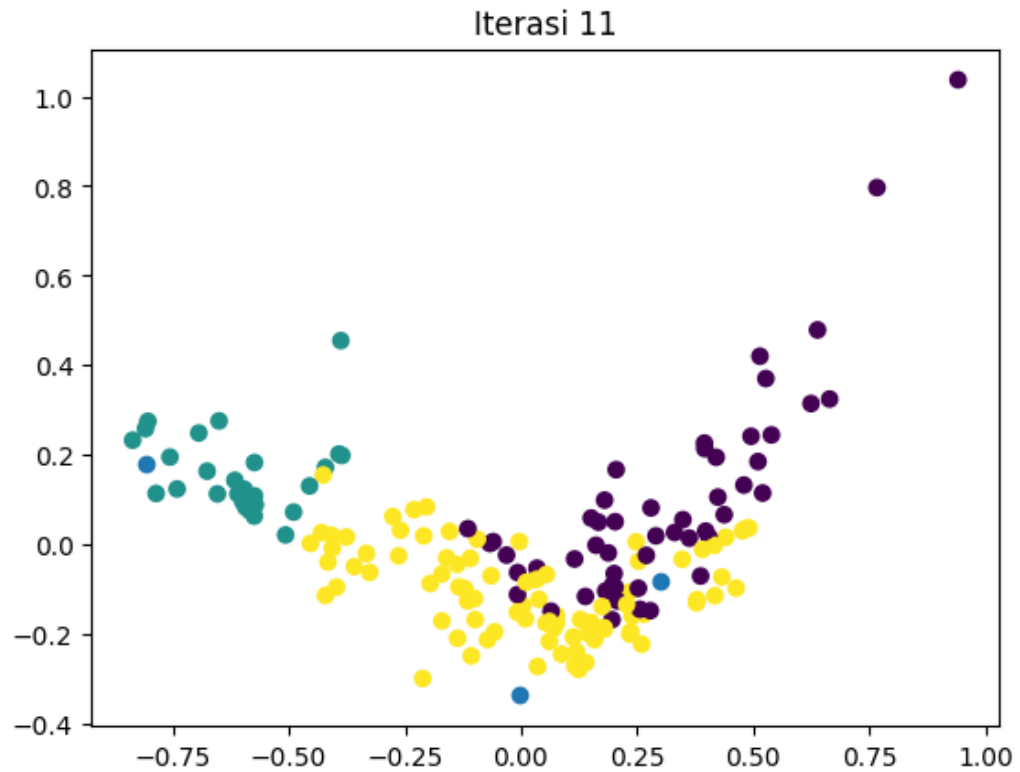
c. Visualisasi data

Pada tahap ini data divisualisasikan dalam bentuk 2 dimensi. Metode PCA (Principal Component Analysis) digunakan untuk mengkonversi atau mereduksi data dengan dimensi yang banyak menjadi sejumlah dimensi yang diinginkan. Pada kasus ini, data dikonversi dari 9 dimensi menjadi 2 dimensi. Data 2 dimensi itu merepresentasikan x dan y pada grafik.

```
def plot_clusters(data, labels, centroids, iteration):  
    pca = PCA(n_components=2)  
    data_2d = pca.fit_transform(data)  
    centroids_2d = pca.transform(centroids.T)  
    clear_output(wait=True)  
    plt.title(f'Iterasi {iteration}')  
    plt.scatter(x=data_2d[:,0], y=data_2d[:,1], c=labels)  
    plt.scatter(x=centroids_2d[:,0], y=centroids_2d[:,1])  
    plt.show()
```

[332] ✓ 0.6s

PCA diterapkan pada tahap ini untuk kebutuhan visualisasi data 2 dimensi.



Contoh implementasi visualisasi data.

d. Pemberian label kelas dan negara

Pada tahap ini, setelah mendapatkan label terakhir pada data, maka data perhitungan, label, dan negara disatukan

```
df_scale_with_label['country'] = df['country']
df_scale_with_label
```

[336] ✓ 0.1s

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp	label	country
0	0.426485	0.049482	0.358608	0.257765	0.008047	0.126144	0.475345	0.736593	0.003073	1	Afghanistan
1	0.068160	0.139531	0.294593	0.279037	0.074933	0.080399	0.871795	0.078864	0.036833	2	Albania
2	0.120253	0.191559	0.146675	0.180149	0.098809	0.187691	0.875740	0.274448	0.040365	2	Algeria
3	0.566699	0.311125	0.064636	0.246266	0.042535	0.245911	0.552268	0.790221	0.031488	1	Angola
4	0.037488	0.227079	0.262275	0.338255	0.148652	0.052213	0.881657	0.154574	0.114242	0	Antigua and Barbuda
...
162	0.129503	0.232582	0.213797	0.302609	0.018820	0.063118	0.609467	0.370662	0.026143	2	Vanuatu
163	0.070594	0.142032	0.192666	0.100809	0.127750	0.463081	0.854043	0.208202	0.126650	2	Venezuela
164	0.100779	0.359651	0.312617	0.460715	0.031200	0.150725	0.808679	0.126183	0.010299	0	Vietnam
165	0.261441	0.149536	0.209447	0.197397	0.031120	0.257000	0.698225	0.555205	0.010299	2	Yemen
166	0.391918	0.184556	0.253574	0.177275	0.021473	0.168284	0.392505	0.670347	0.011731	1	Zambia

167 rows × 11 columns

Bagian 4

Evaluasi Hasil

Pada metode kMeans ini, terdapat parameter yang diinputkan, yaitu jumlah k atau banyaknya kelas yang akan dibuat. Cara untuk mencari nilai k yang tepat salah satunya adalah menggunakan metode elbow.

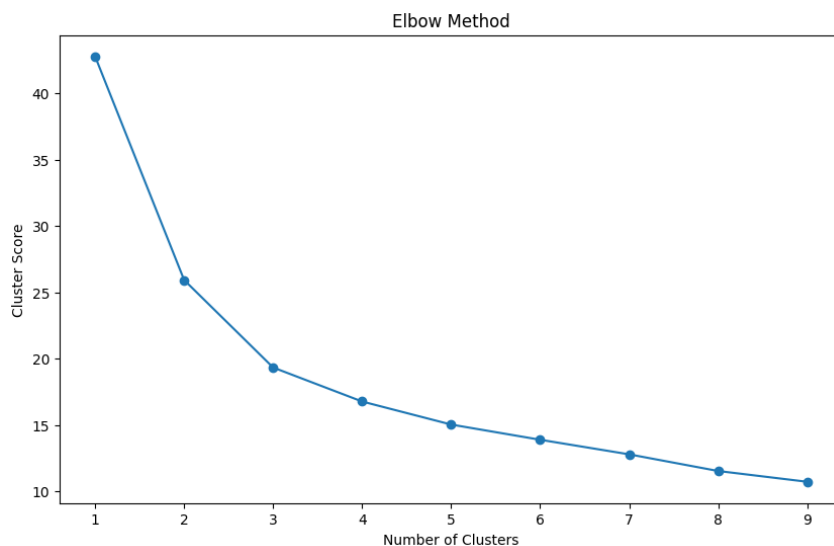
```
[337] ✓ 0.9s

from sklearn.cluster import KMeans

cluster_score = []
for i in range(1, 10):
    kmeans = KMeans(n_clusters = i, init = 'random', random_state = 42)
    kmeans.fit(df_scaling)
    cluster_score.append(kmeans.inertia_)

plt.figure(figsize=(10,6))
plt.plot(range(1, 10), cluster_score, marker = 'o')
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('Cluster Score')
plt.show()

[338] ✓ 1.9s
```



Hasil dari metode elbow adalah banyaknya cluster yang tepat adalah 3 cluster.

Bagian 5

Presentasi Video

Berikut ini adalah kumpulan resource yang ada pada tugas ini

https://drive.google.com/drive/folders/1j9OISrGEG79shI4udK7nxTMXRuBKGT6?usp=share_link