

Final Project

Clustering the Countries by using K-Means for HELP International

by : Ramadito Ferdian Assa



OBJECTIVE

Untuk mengkategorikan negara menggunakan faktor sosial ekonomi dan kesehatan yang menentukan pembangunan negara secara keseluruhan.

TENTANNG ORGANISASI

HELP International adalah LSM kemanusiaan internasional yang berkomitmen untuk memerangi kemiskinan dan menyediakan fasilitas dan bantuan dasar bagi masyarakat di negara-negara terbelakang saat terjadi bencana dan bencana alam.

PERMASALAHAN

HELP International telah berhasil mengumpulkan sekitar \$ 10 juta. Saat ini, CEO LSM perlu memutuskan bagaimana menggunakan uang ini secara strategis dan efektif. Jadi, CEO harus mengambil keputusan untuk memilih negara yang paling membutuhkan bantuan. Oleh karena itu, Tugas teman-teman adalah mengkategorikan negara menggunakan beberapa faktor sosial ekonomi dan kesehatan yang menentukan perkembangan negara secara keseluruhan. Kemudian kalian perlu menyarankan negara mana saja yang paling perlu menjadi fokus CEO.

PENJELASAN KOLOM FITUR :

- Negara : Nama negara
- Kematian_anak: Kematian anak di bawah usia 5 tahun per 1000 kelahiran
- Ekspor : Ekspor barang dan jasa perkapita
- Kesehatan: Total pengeluaran kesehatan perkapita
- Impor: Impor barang dan jasa perkapita
- Pendapatan: Penghasilan bersih perorang
- Inflasi: Pengukuran tingkat pertumbuhan tahunan dari Total GDP
- Harapan_hidup: Jumlah tahun rata-rata seorang anak yang baru lahir akan hidup jika pola kematian saat ini tetap sama
- Jumlah_fertiliti: Jumlah anak yang akan lahir dari setiap wanita jika tingkat kesuburan usia saat ini tetap sama
- GDPperkapita: GDP per kapita. Dihitung sebagai Total GDP dibagi dengan total populasi.

Library Python

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from sklearn.cluster import KMeans
```



Dataframe Read

```
df = pd.read_csv('Data_Negara_HELP.csv')
df.head()

✓ 0.3s
```

	Negara	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita
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

Dataframe Read

```
df.info()
✓ 0.3s

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Negara            167 non-null    object  
 1   Kematian_anak     167 non-null    float64 
 2   Ekspor             167 non-null    float64 
 3   Kesehatan          167 non-null    float64 
 4   Impor              167 non-null    float64 
 5   Pendapatan         167 non-null    int64   
 6   Inflasi             167 non-null    float64 
 7   Harapan_hidup      167 non-null    float64 
 8   Jumlah_fertiliti    167 non-null    float64 
 9   GDPperkapita        167 non-null    int64   
dtypes: float64(7), int64(2), object(1)
memory usage: 13.2+ KB
```

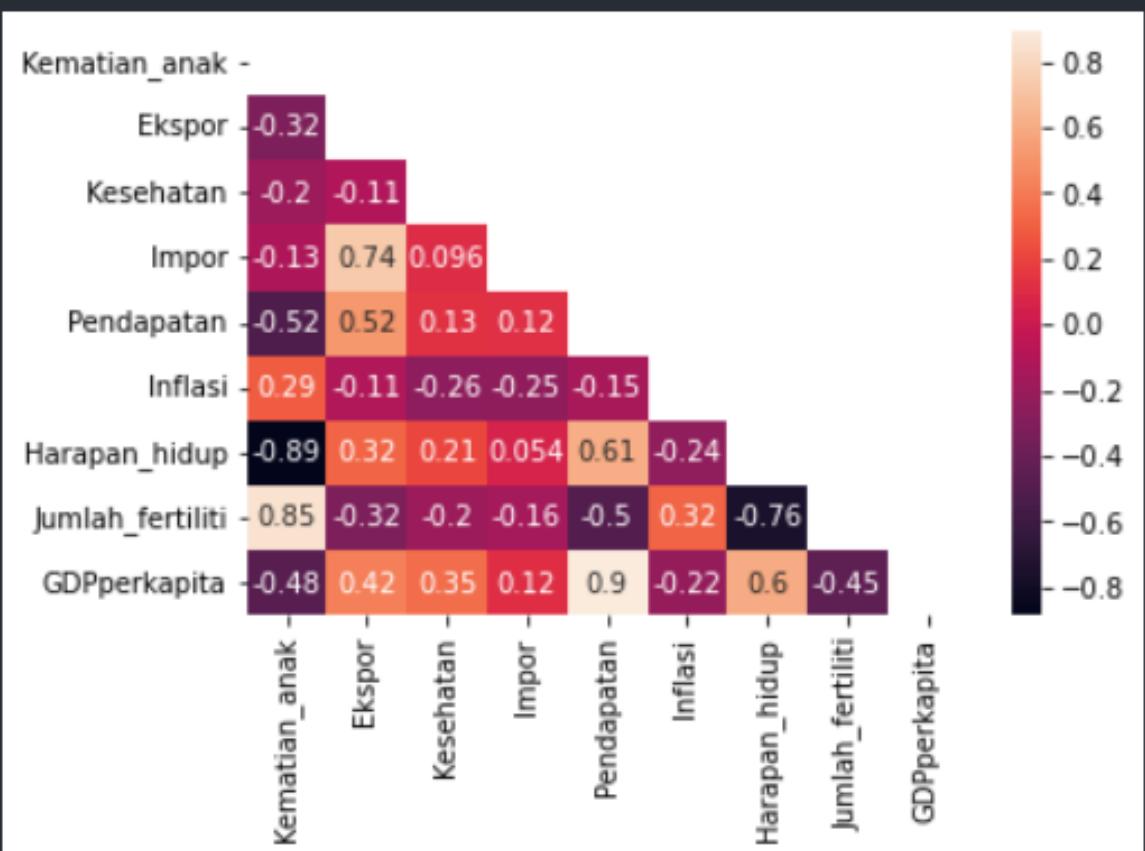
Dataframe memiliki 167 Rows dan 10 Columns

Multivariate Analysis with Heatmap

```
sns.heatmap(df.corr(), mask=np.triu(np.ones_like(df.corr())), annot=True)  
✓ 0.2s
```

Python

<AxesSubplot:>



```
df_new = df[['Negara', 'Kematian_anak', 'Pendapatan']]
```

```
✓ 0.4s
```

	Negara	Kematian_anak	Pendapatan
0	Afghanistan	90.2	1610
1	Albania	16.6	9930
2	Algeria	27.3	12900
3	Angola	119.0	5900
4	Antigua and Barbuda	10.3	19100
...
162	Vanuatu	29.2	2950
163	Venezuela	17.1	16500
164	Vietnam	23.3	4490
165	Yemen	56.3	4480
166	Zambia	83.1	3280

167 rows × 3 columns

```
sns.pairplot(df, corner=True);  
✓ 0.2s
```

Python

Berdasarkan heatmap, kami memutuskan mengevaluasi kolom 'Kematian_anak' dan 'Pendapatan' dikarenakan memiliki korelasi negatif yang cukup tinggi dan juga kedua kolom tersebut relevan dengan tujuan yaitu menentukan Negara mana yang perlu menerima bantuan berdasarkan tingkat Ekonomi dan Kesehatan



Data Cleaning

1. Missing Value
2. Handling Outliers

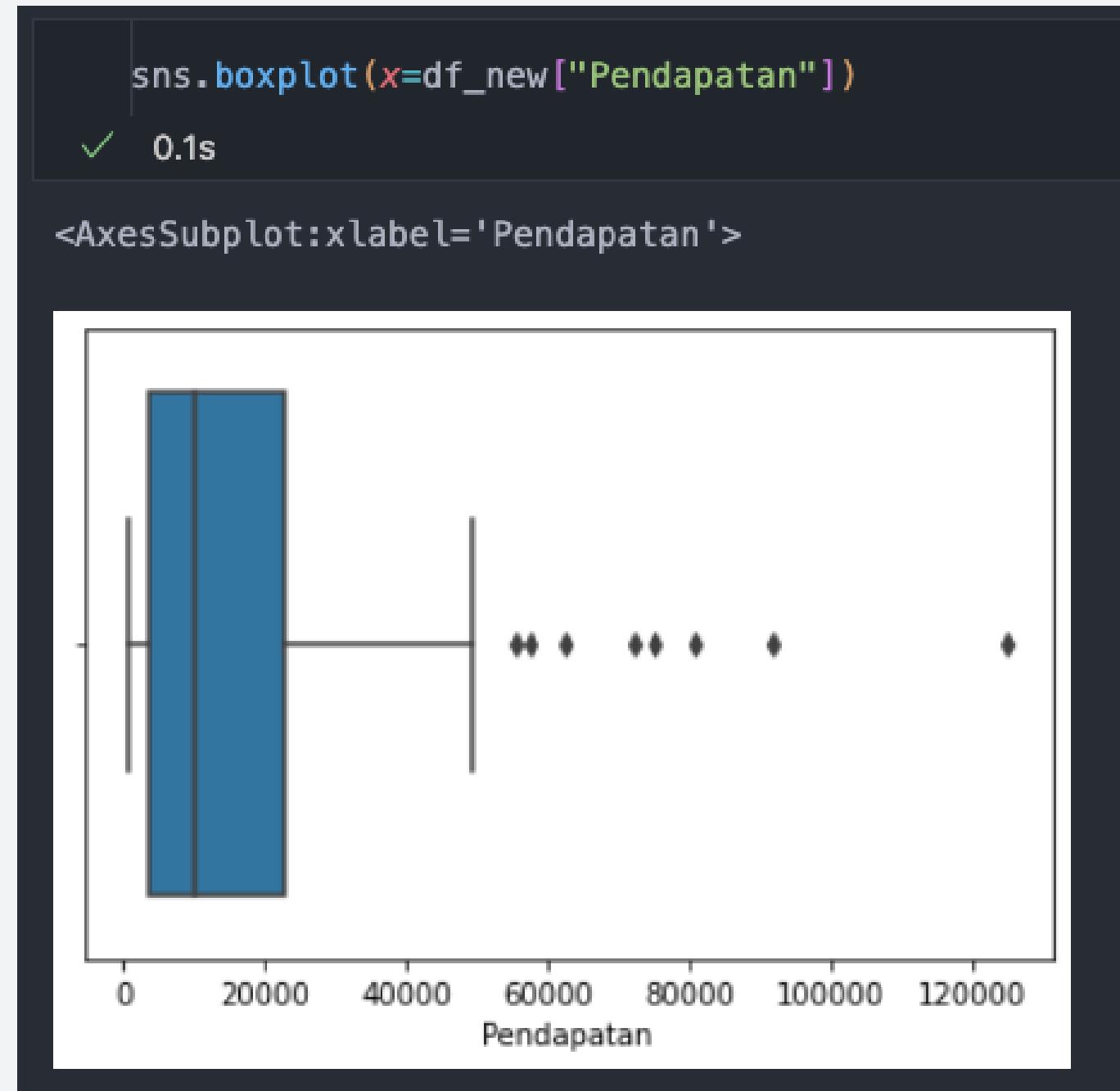
Missing Value

```
| df_new.info()
| ✓ 0.3s
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Negara            167 non-null    object  
 1   Kematian_anak    167 non-null    float64 
 2   Pendapatan        167 non-null    int64   
dtypes: float64(1), int64(1), object(1)
memory usage: 4.0+ KB
Tidak ditemukan missing value pada dataframe
```

Handling Outliers

Pada kasus ini kami tidak melakukan penghapusan pada data yang merupakan outliers, namun kami mengganti value yang merupakan outliers dengan mean dari data tersebut.

Handling Outliers



Terdapat outliers dari kedua fitur diatas

Clustering



1. Scalling Data with MinMaxScaler

```
# Rescaling data dengan MinMaxScaler  
  
scaler = MinMaxScaler()  
  
df_new[['Kematian_anak', 'Pendapatan']] = scaler.fit_transform(df_new[['Kematian_anak', 'Pendapatan']])  
  
df_new.describe()  
✓ ✓ 0.3s
```

/var/folders/br/dnqls7751tdb97g6z28kqgxw0000gn/T/ipykernel_94418/530802698.py:5: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.

Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
df_new[['Kematian_anak', 'Pendapatan']] = scaler.fit_transform(df_new[['Kematian_anak', 'Pendapatan']])
```

	Kematian_anak	Pendapatan
count	167.000000	167.000000
mean	0.363488	0.393818
std	0.238901	0.252358
min	0.000000	0.000000
25%	0.154372	0.155220
50%	0.363488	0.393818
75%	0.592511	0.533195
max	1.000000	1.000000

```
df_new_ready = df_new[['Kematian_anak', 'Pendapatan']]  
✓ 0.2s
```

2. Decide the number of clusters : Elbow Method

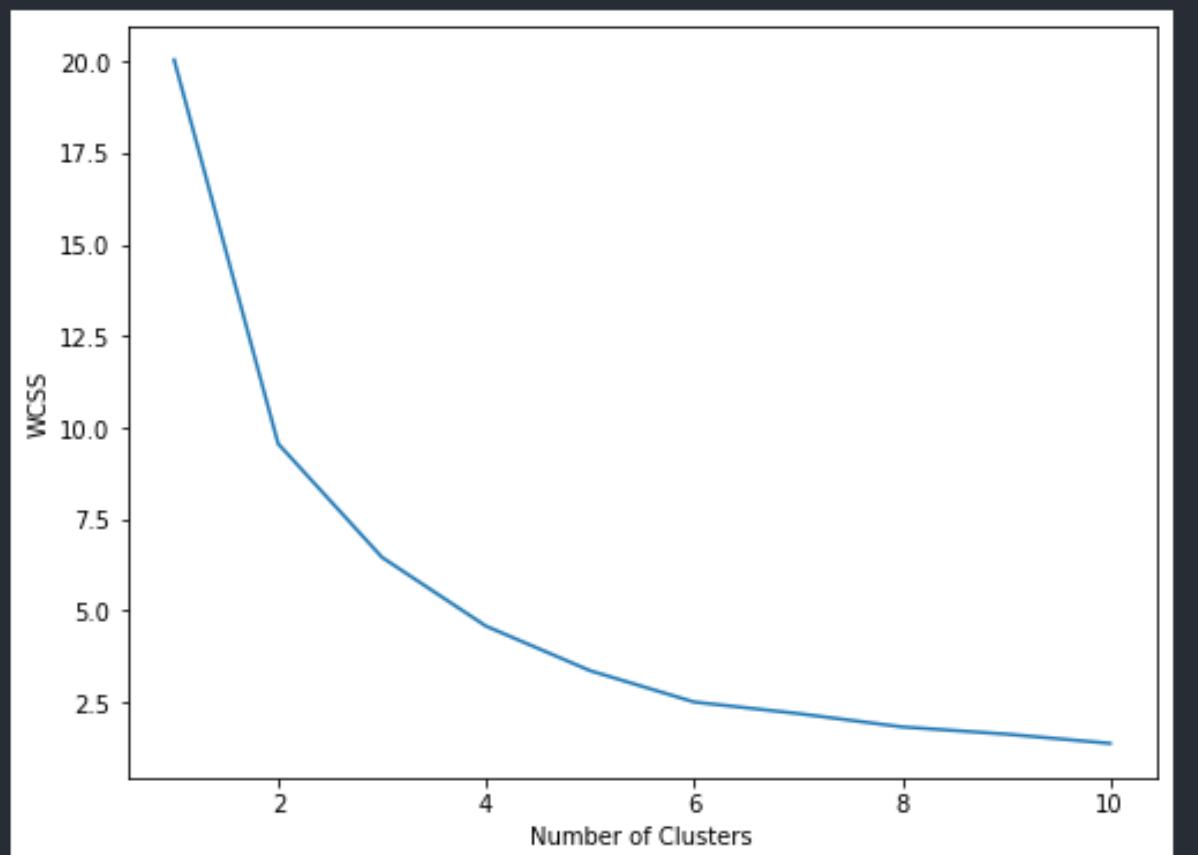
```
sse = []
k_list = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters = k).fit(X)
    centroids = kmeans.cluster_centers_
    prediksi = kmeans.predict(X)
    nilai_sse = 0

    for i in range(len(X)):
        titik_pusat = centroids[prediksi[i]]
        nilai_sse += (X[i, 0] - titik_pusat[0]) ** 2 + (X[i, 1] - titik_pusat[1]) ** 2

    sse.append(nilai_sse)
    k_list.append(k)

plt.figure(figsize=(8,6))
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.plot(k_list, sse)
plt.show()
```

✓ 0.2s



3. Clustering with K = 3

```
df_new_ready.shape

kmeans = KMeans(n_clusters=3)

label = kmeans.fit_predict(df_new_ready)

print(label)
✓ 0.4s

[1 2 2 1 0 0 1 0 0 2 0 0 1 2 0 0 1 1 1 1 0 2 2 0 2 1 1 1 1 0 1 1 1 0 2 2 1
 1 1 0 1 0 0 0 0 2 2 2 1 2 1 0 1 0 0 2 1 1 0 1 0 2 1 1 1 1 1 0 0 1 1 2 2 0
 0 0 1 0 2 1 1 1 0 1 1 2 2 1 1 1 0 0 0 1 1 0 0 1 0 1 2 1 1 1 0 1 1 1 0 1 0
 0 1 1 0 0 1 2 1 2 1 0 0 0 2 0 1 1 2 1 0 0 1 0 0 0 1 2 0 0 0 2 1 2 0 0 1 1
 2 1 1 1 2 2 2 1 0 0 0 0 2 1 1 2 1 1 1]
```

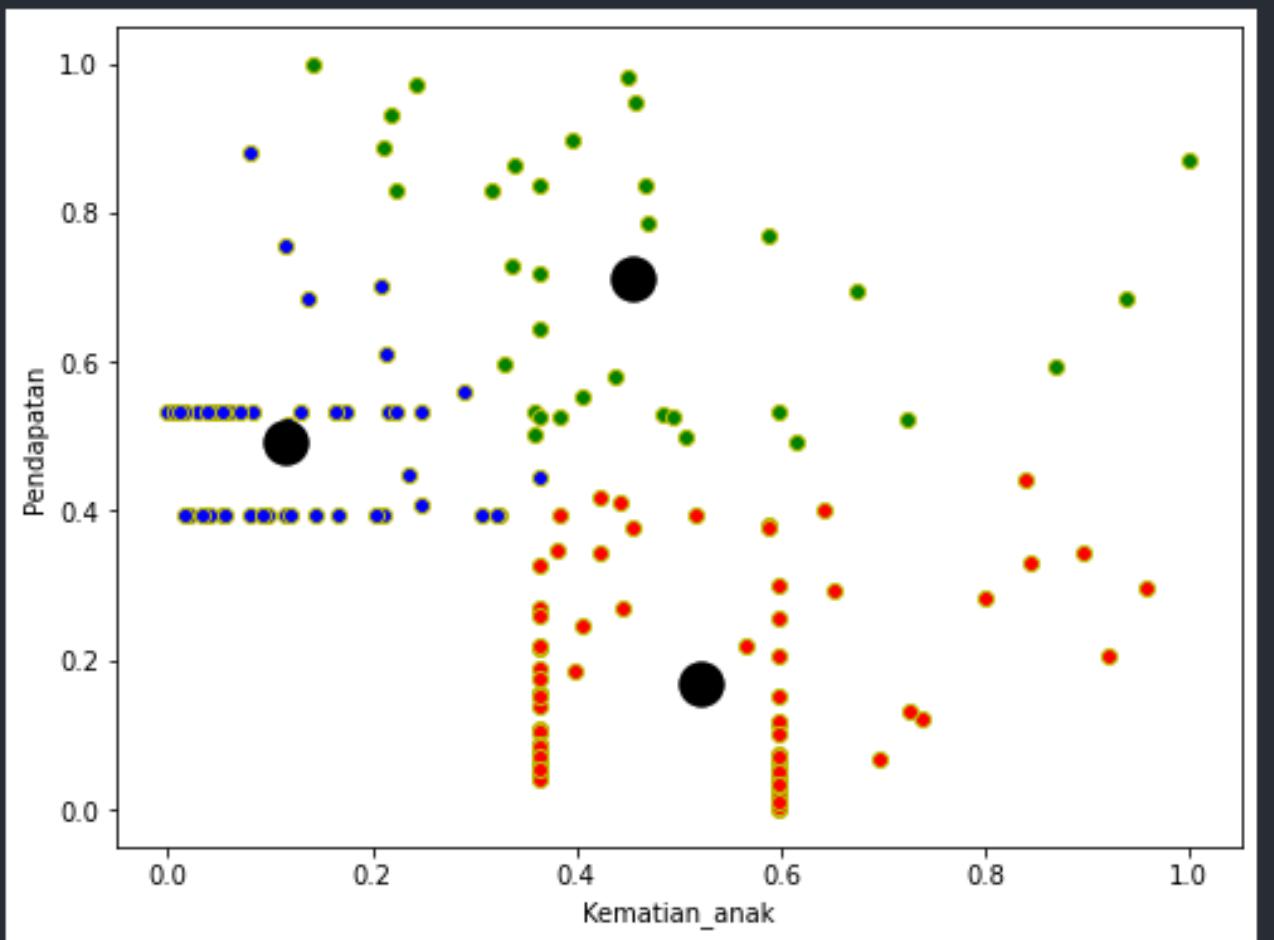
4. Visualize clustering

```
plt.figure(figsize=(8,6))
plt.xlabel('Kematian_anak')
plt.ylabel('Pendapatan')

plt.scatter(X[label==0, 0], X[label==0, 1], c='blue', label ='Cluster 0', edgecolors='y')
plt.scatter(X[label==1, 0], X[label==1, 1], c='red', label ='Cluster 1',edgecolors='y')
plt.scatter(X[label==2, 0], X[label==2, 1], c='green', label ='Cluster 2',edgecolors='y')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=300, c='black', label = 'Centroids')
```

✓ 0.1s

<matplotlib.collections.PathCollection at 0x7fce62851910>



Recomendation

```
cluster_1 = df_new["Cluster"] == 1  
df_new[cluster_1]  
✓ 0.4s
```

	Negara	Kematian_anak	Pendapatan	Kematian_anak_Outlier	Pendapatan_Outlier	Cluster
0	Afghanistan	0.597591	0.056582	False	False	1
3	Angola	0.597591	0.299079	False	False	1
6	Armenia	0.423497	0.344299	False	False	1
12	Bangladesh	0.363488	0.103499	False	False	1
16	Belize	0.442623	0.411000	False	False	1
...
161	Uzbekistan	0.920765	0.205246	False	False	1
162	Vanuatu	0.726776	0.132327	False	False	1
164	Vietnam	0.565574	0.219377	False	False	1
165	Yemen	0.363488	0.218812	False	False	1
166	Zambia	0.363488	0.150981	False	False	1

74 rows × 6 columns

```
df_sorted = df_new[['Negara', 'Kematian_anak', 'Pendapatan']]  
df_sorted = df_new[cluster_1].sort_values(['Kematian_anak', 'Pendapatan'], ascending = [False, True])  
df_sorted.drop(['Kematian_anak_Outlier', 'Pendapatan_Outlier'], axis=1, inplace=True)  
✓ ✓ 0.7s
```

Rekomendasi berdasarkan tingkat Kematian Anak tinggi dan tingkat Pendapatan rendah

```
df_sorted.head()
```

✓ 0.1s

	Negara	Kematian_anak	Pendapatan	Cluster
65	Guyana	0.956284	0.295687	1
161	Uzbekistan	0.920765	0.205246	1
62	Guatemala	0.896175	0.344865	1
105	Morocco	0.844262	0.329603	1
70	Indonesia	0.838798	0.442089	1