HELP INTERNATIONAL

Final Project

By

Kelvin Erlangga
kelvinerlangga2002@gmail.com

UNDERSTANDING THE PROBLEM

Problem Descriptions (In Indonesian)

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 pada proyek kali ini adalah mengkategorikan negara menggunakan beberapa faktor sosial ekonomi dan kesehatan yang menentukan perkembangan negara secara keseluruhan. Kemudian, tentukan negara mana saja yang paling perlu menjadi fokus CEO.

Goals

Create KMean clusters of countries and select the most suitable countries for financial aid from HELP international by following the following criterias for underdeveloped countries : highest child mortality, lowest export, lowest health, highest import, lowest income, highest inflation, lowest life expectancy, highest total fertility, lowest GDPP.

DATA DICTIONARY

Data Descriptions (In Indonesian)

- 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

Steps

01

Reading & Understanding Data

02

Exploratory Data Analysis

Data Cleansing,
Univariate, Bivariate, and
Multivariate Analysis

03

Outliers Treatment

Steps

04

Scaling the Data

05

Clustering and their Visualization

06

Report Countries

Reading and Understanding the Data

01

Import Necessary Libraries

```
[ ] # Ignore warnings
    import warnings
    warnings.filterwarnings('ignore')
```

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

    # For Visualisation
    import matplotlib.pyplot as plt
    import seaborn as sns
    %matplotlib inline

    # For rescaling the data
    from sklearn.preprocessing import StandardScaler

    # To perform KMeans clustering
    from sklearn.cluster import KMeans

    # To check most accurate KMeans clustering
    from sklearn.metrics import silhouette_score
```

Libraries that Are Used

1. Warnings (to ignore any warnings)
2. Numpy and Pandas (for data analysis)
3. Matplotlib and Seaborn (for data visualization)
4. Scikit-learn (for Machine Learning)

Reading the Data

```
[ ] # Check the top 5 rows of dataframe
    df = pd.read_csv('https://drive.google.com/uc?export=download&id=1O6sws-MvbZ1rG2k2izn-DTseKduqiK8j')
    df.head()
```

	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

```
[ ] # Check the bottom 5 rows of dataframe
    df.tail()
```

	Negara	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita
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

Inspecting the Data

```
# Check number of non-null data from each column and their datatype
df.info()
```

```
<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
```

```
# Check number of unique data in each columns
df.nunique()
```

```
Negara              167
Kematian_anak       139
Ekspor              147
Kesehatan           147
Impor               151
Pendapatan          156
Inflasi             156
Harapan_hidup       127
Jumlah_fertiliti    138
GDPperkapita        157
dtype: int64
```

Insights

1. The shape of data is (167, 10)
2. The dataframe does not have any null value
3. Each data of column 'negara' is unique. Thus, the dataframe does not have duplicate data.

02

Exploratory Data Analysis

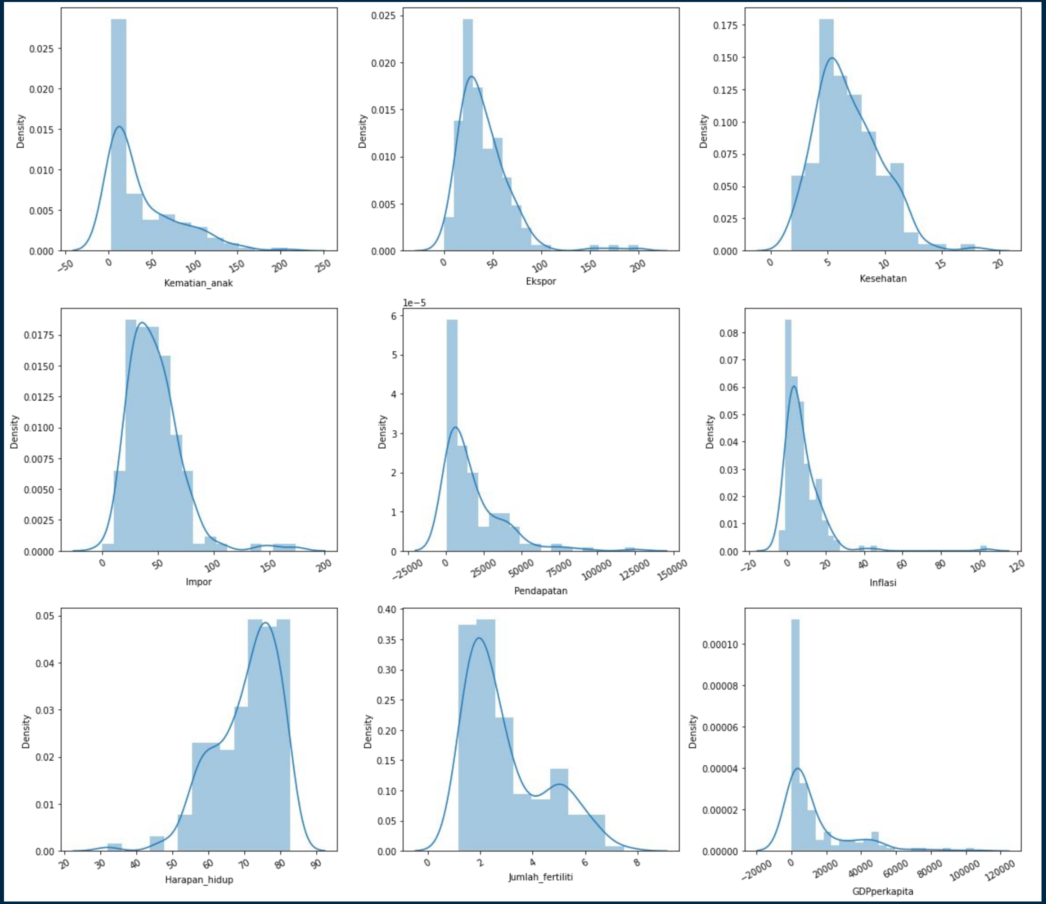
Data Cleansing

```
[ ] # Check any null values from given dataset
    df.isna().sum()

    Negara              0
    Kematian_anak       0
    Ekspor              0
    Kesehatan           0
    Impor               0
    Pendapatan          0
    Inflasi             0
    Harapan_hidup       0
    Jumlah_fertiliti    0
    GDPperkapita        0
    dtype: int64
```

! Informations !

From series beside, we know that the dataframe does not have any null values. Therefore, we can continue to next step.

Univariate Analysis

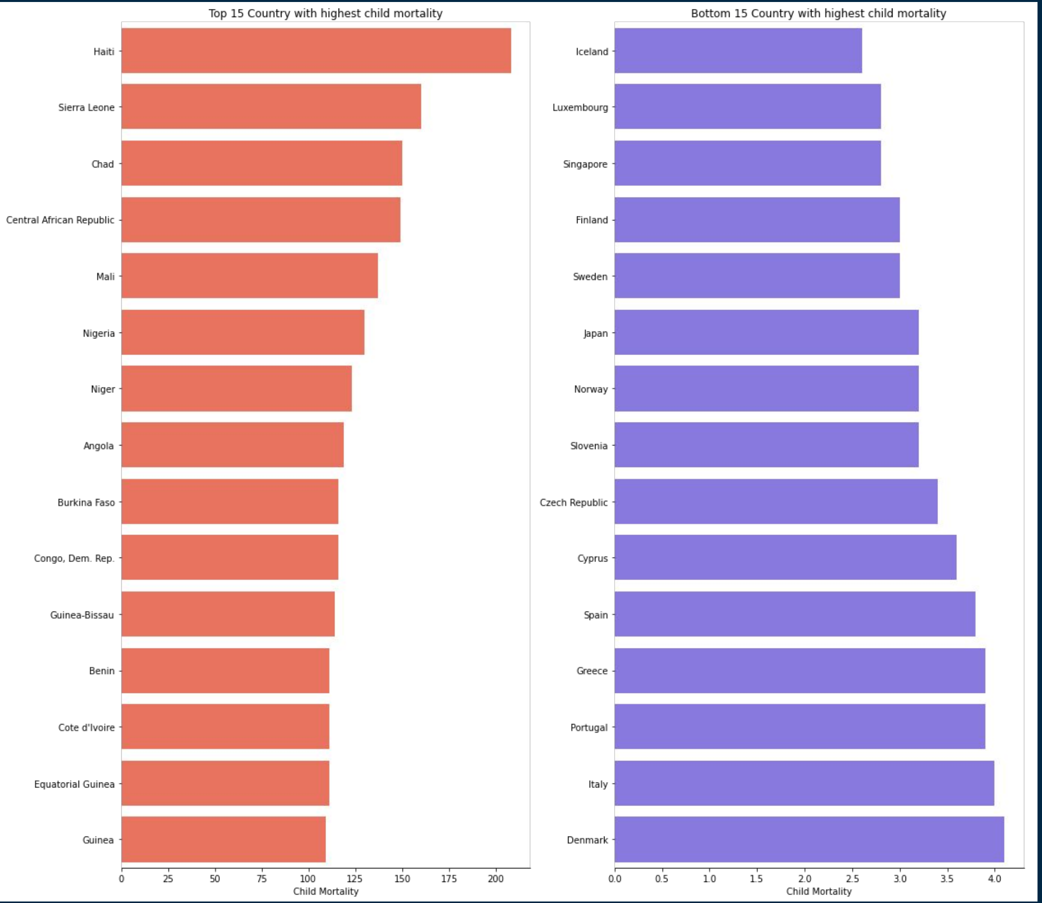


Insights

From distribution plots beside, we can get several insights.

- We can see that there are outliers in the data distribution of each feature.
- We can also see that each distribution plot that represents a feature tends to have a skewness of either right or left skew. The fact that they have a skewness shows that there is a quite large gap between Well-developed countries and Under-developed countries.

Bivariate Analysis (1)

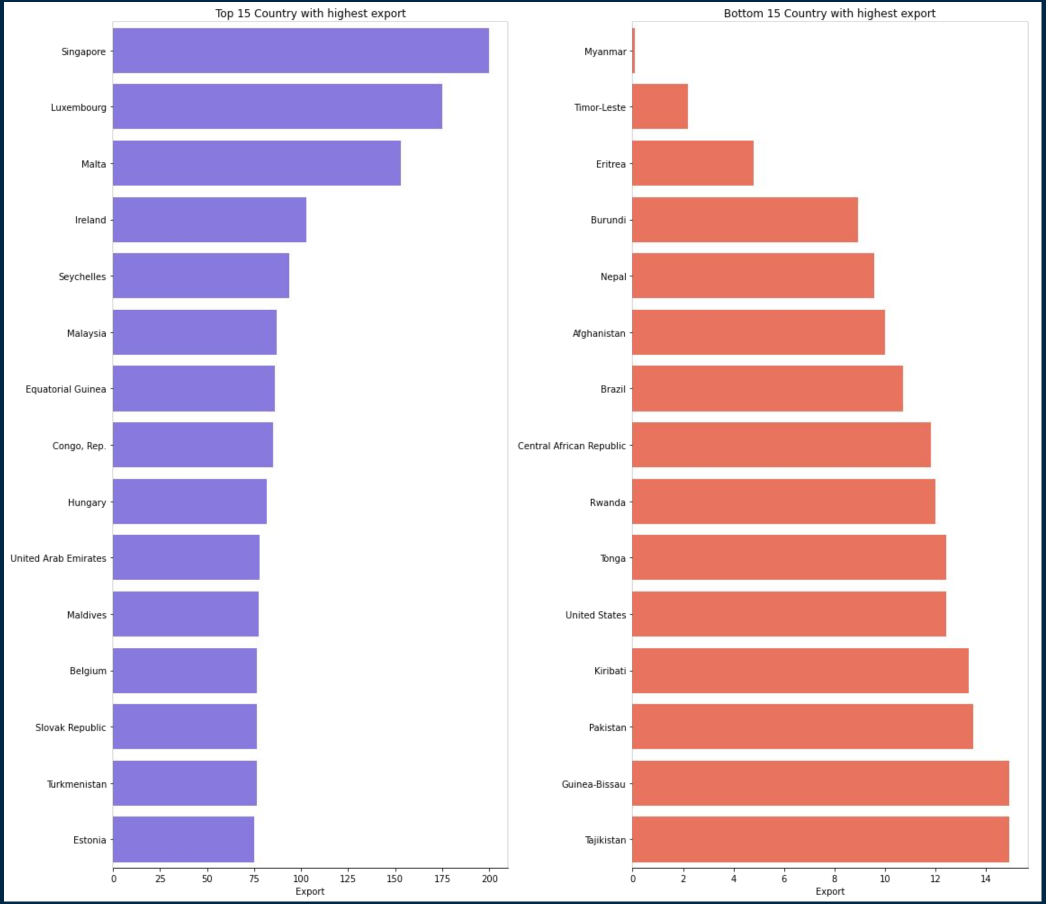


Insights

From bar plots beside, we can get several insights.

- We can see that Haiti has the highest child mortality rate.
- We can also see that Iceland has the lowest child mortality rate.

Bivariate Analysis (2)

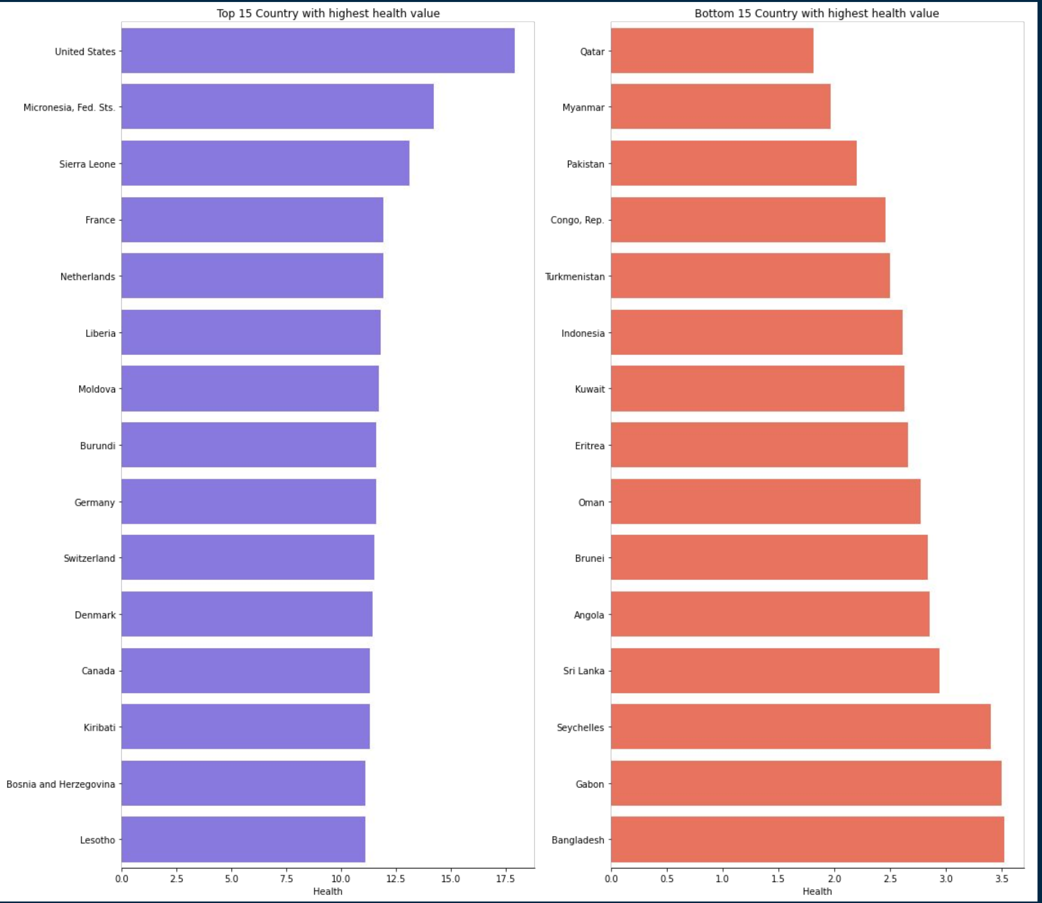


Insights

From bar plots beside, we can get several insights.

- We can see that Singapore has the highest export rate.
- We can also see that Myanmar has the lowest export rate.

Bivariate Analysis (3)

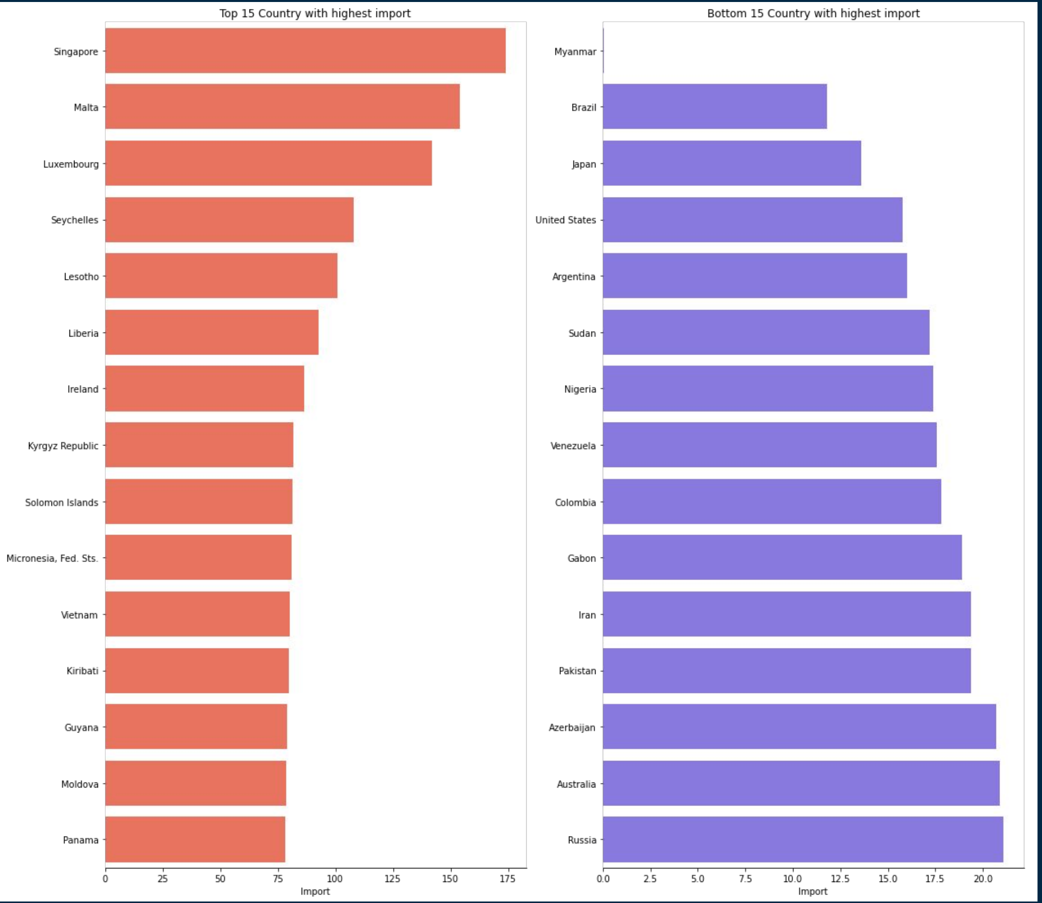


Insights

From bar plots beside, we can get several insights.

- We can see that United States has the highest health rate.
- We can also see that Qatar has the lowest health rate.

Bivariate Analysis (4)

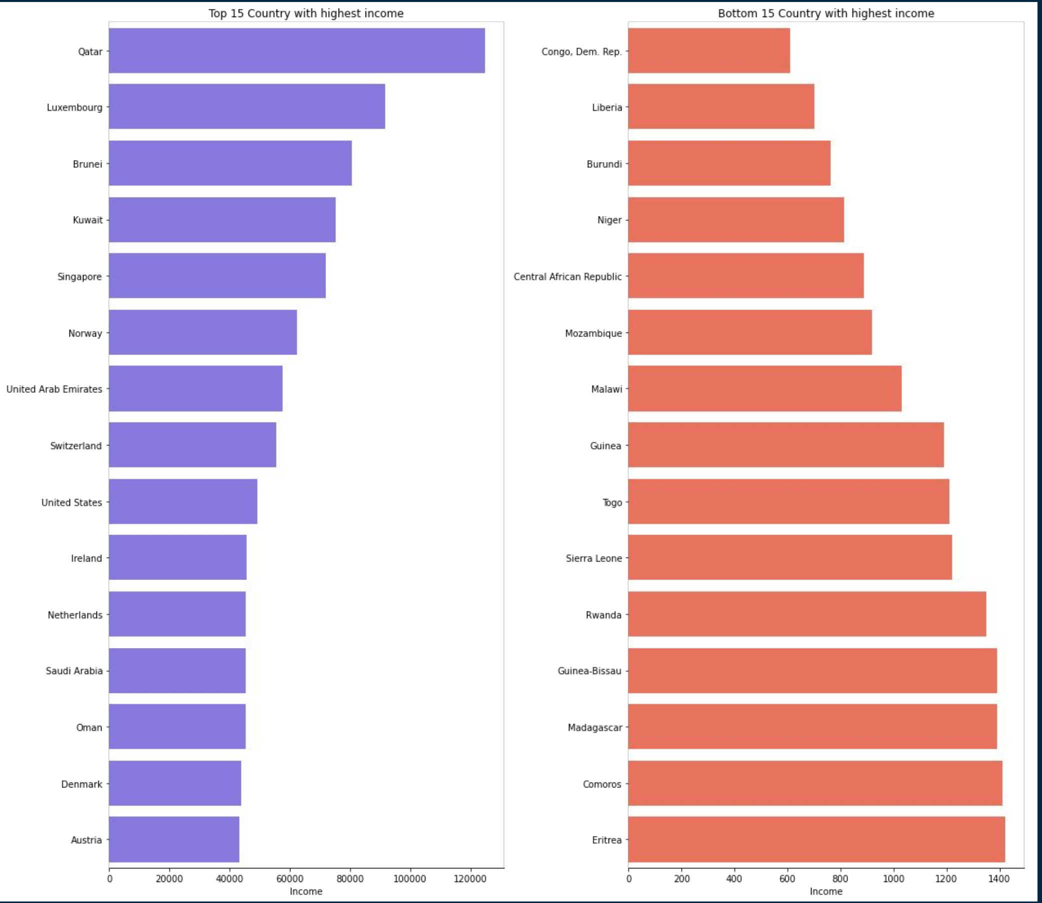


Insights

From bar plots beside, we can get several insights.

- We can see that Singapore has the highest import rate.
- We can also see that Myanmar has the lowest import rate.

Bivariate Analysis (5)

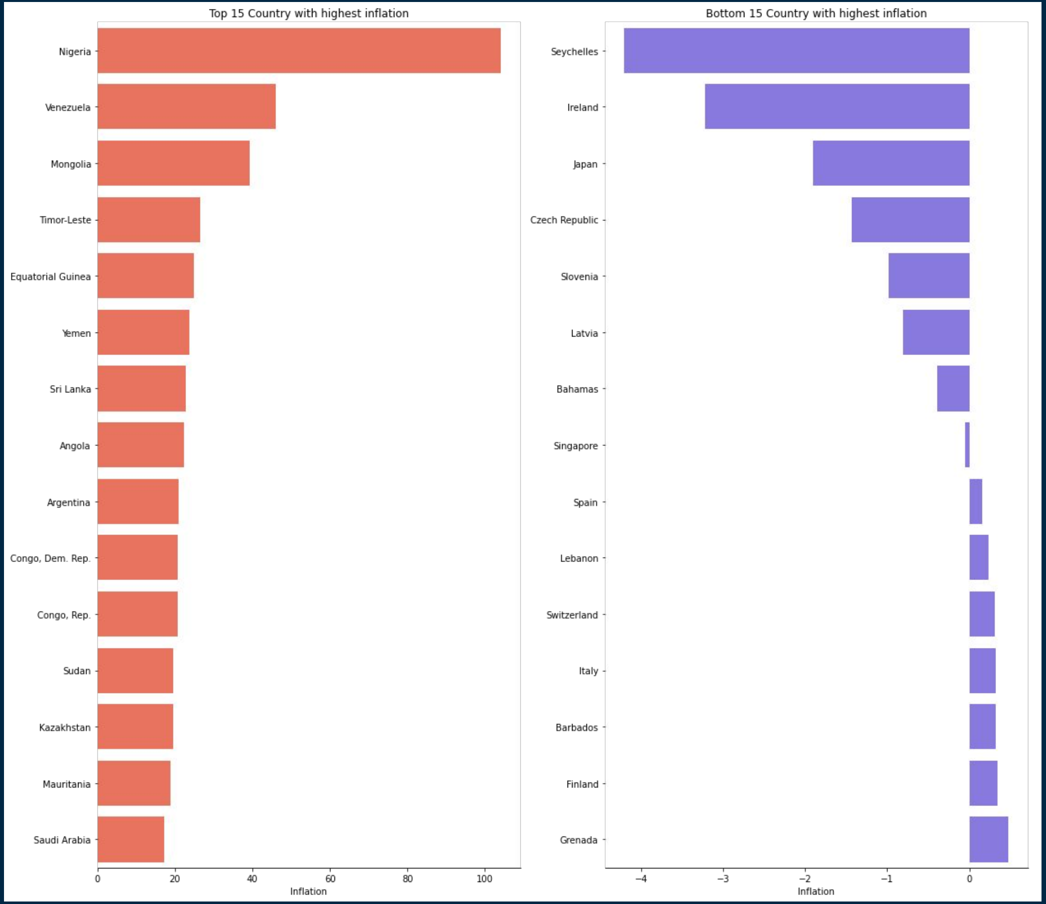


Insights

From bar plots beside, we can get several insights.

- We can see that Qatar has the highest income.
- We can also see that Congo, Dem. Rep. has the lowest income.

Bivariate Analysis (6)

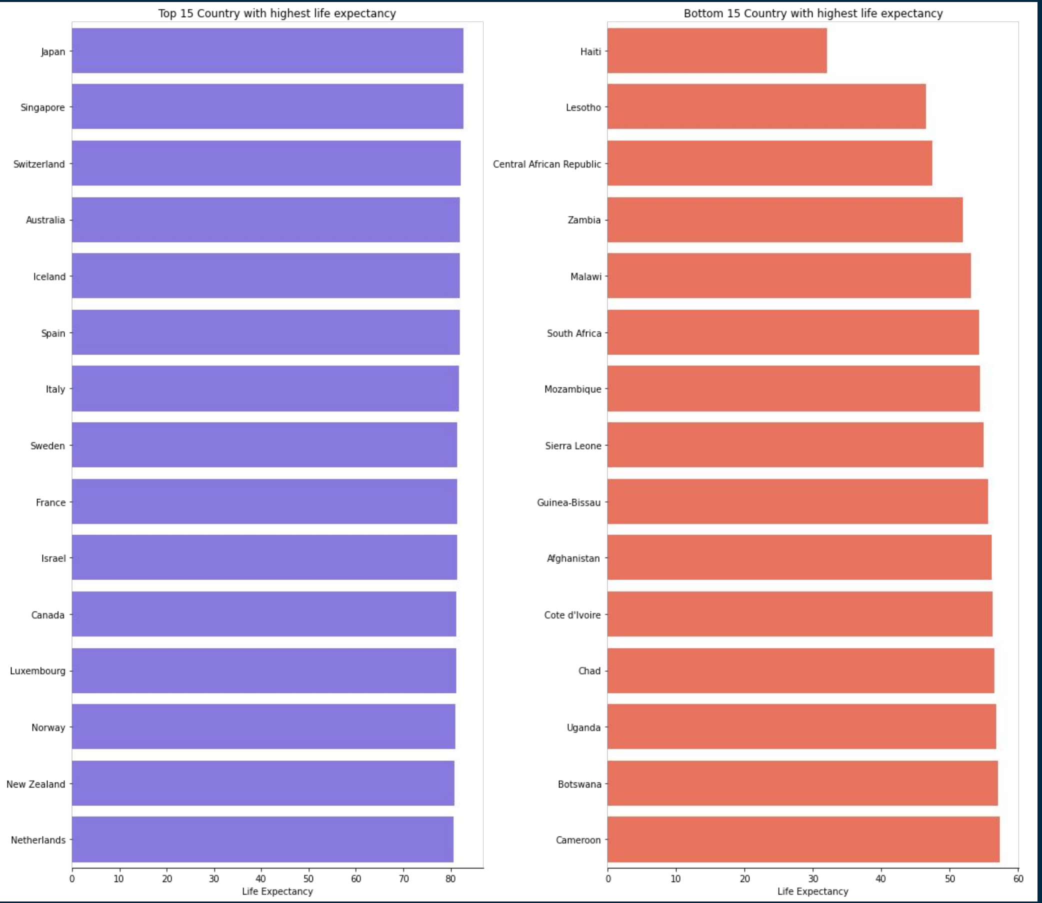


Insights

From bar plots beside, we can get several insights.

- We can see that Nigeria has the highest inflation rate.
- We can also see that Seychelles. has the lowest inflation rate.

Bivariate Analysis (7)

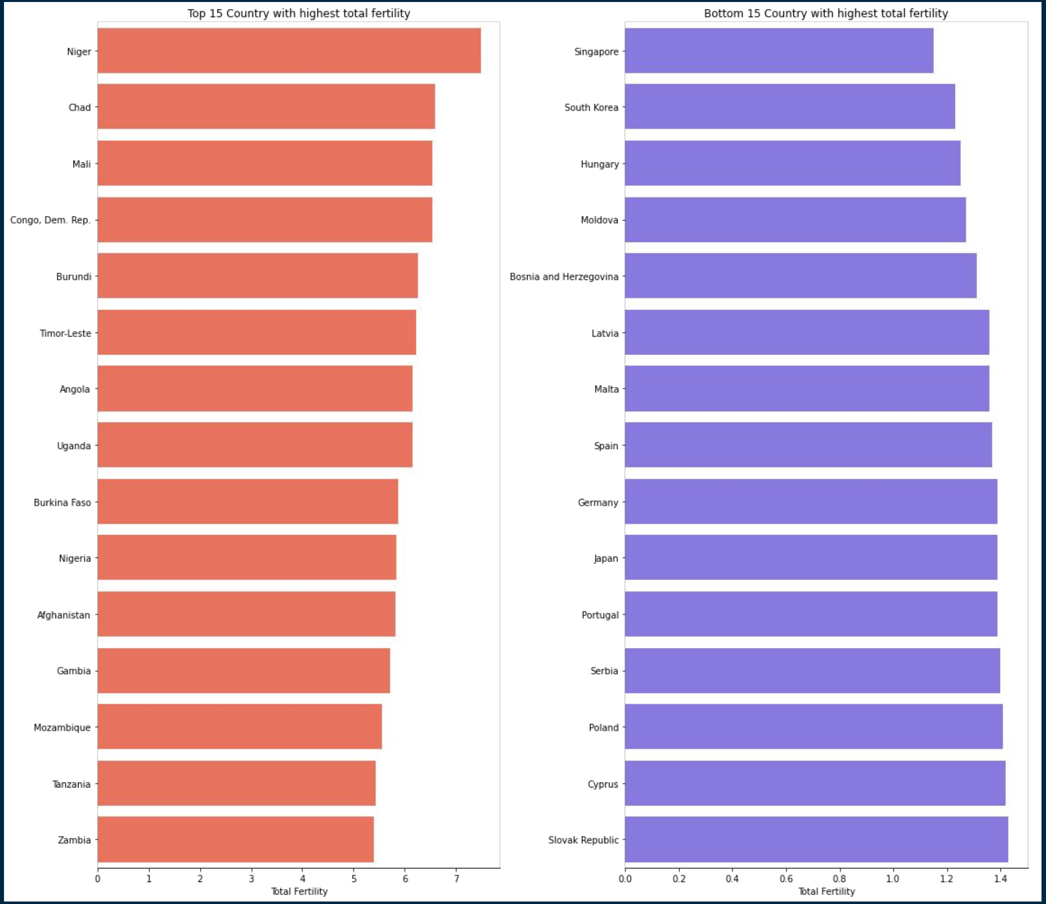


Insights

From bar plots beside, we can get several insights.

- We can see that Japan has the highest life expectancy.
- We can also see that Haiti has the lowest life expectancy.

Bivariate Analysis (8)

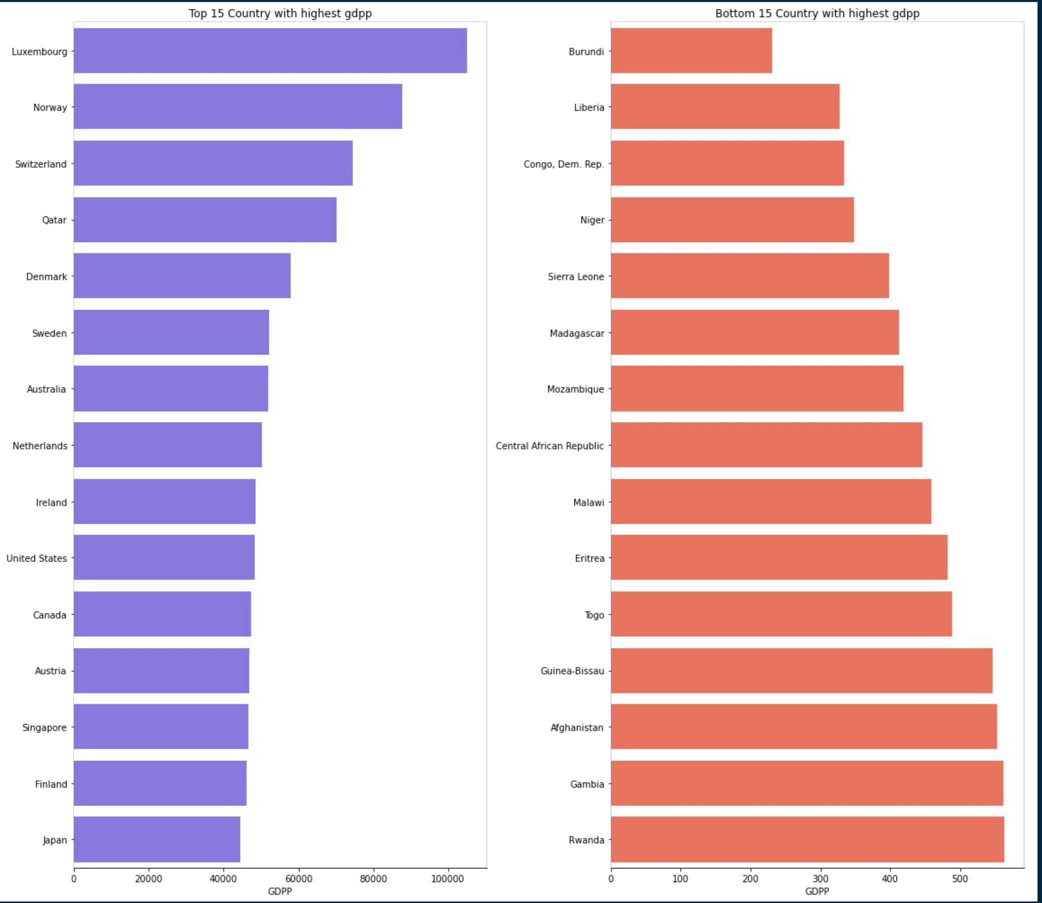


Insights

From bar plots beside, we can get several insights.

- We can see that Niger has the highest total fertility.
- We can also see that Singapore has the lowest total fertility.

Bivariate Analysis (9)

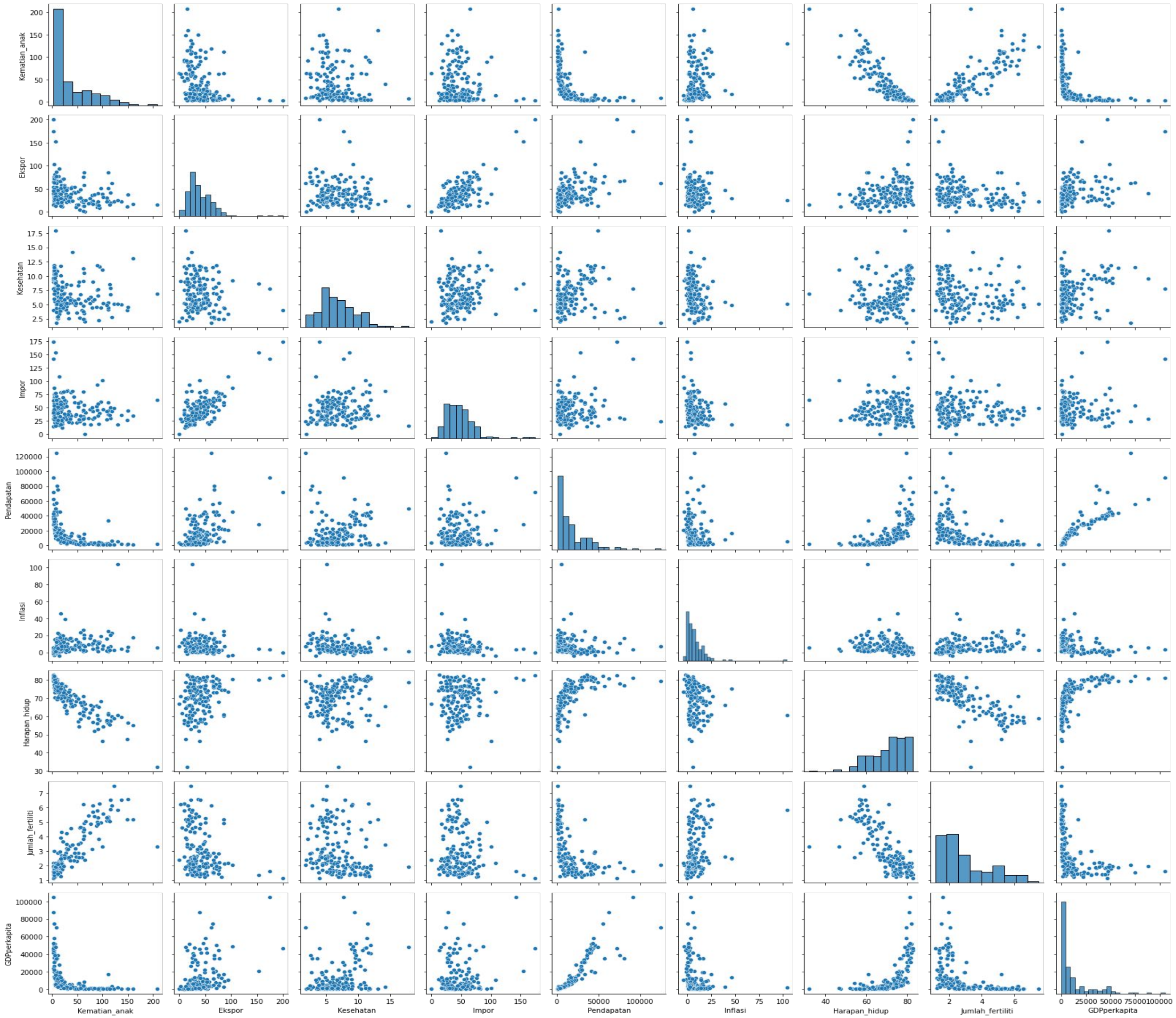


Insights

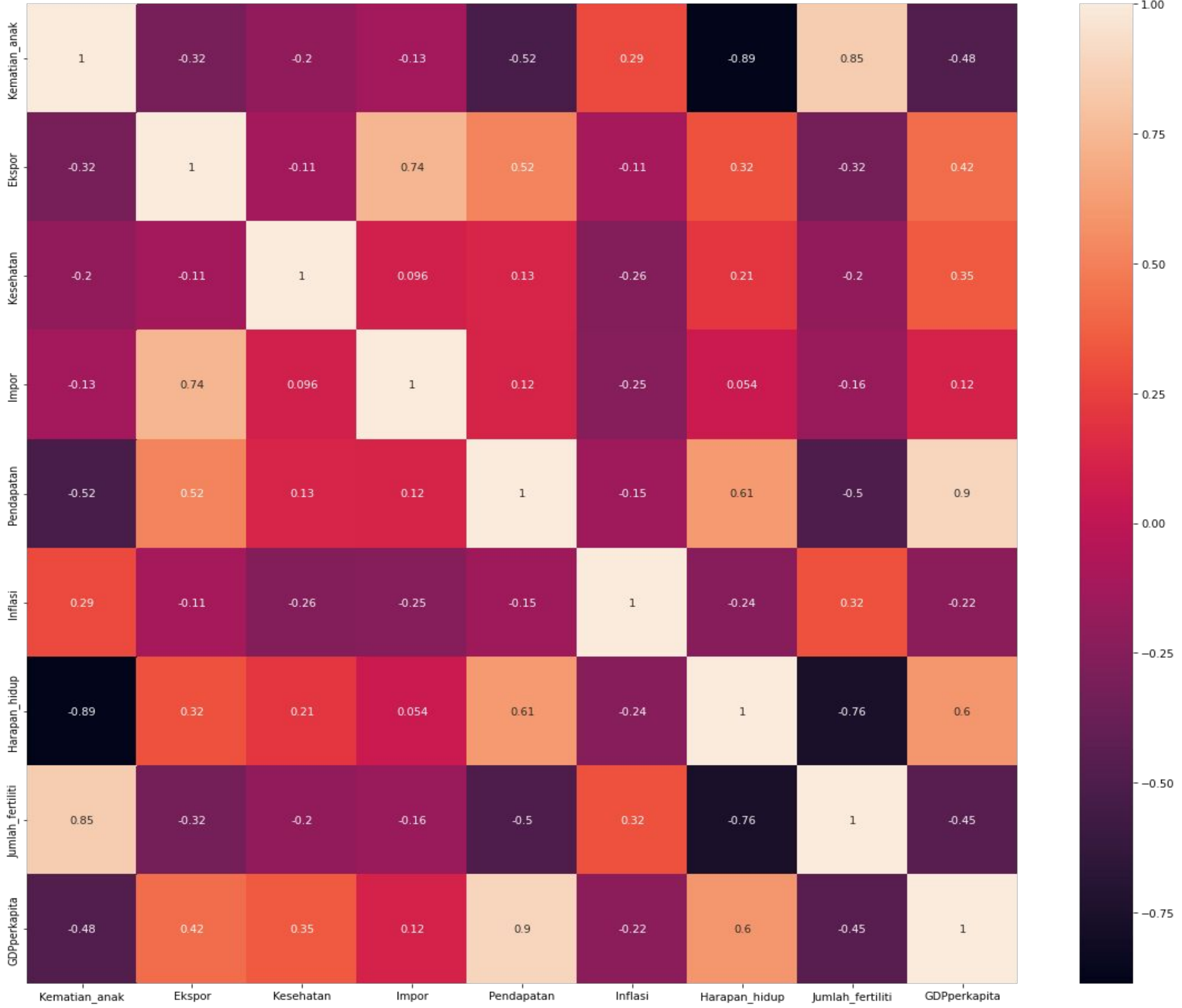
From bar plots beside, we can get several insights.

- We can see that Luxembourg has the highest GDPP.
- We can also see that Burundi has the lowest GDPP.

Multivariate Analysis (Pair Plot)



Multivariate Analysis (Heat Map)



Multivariate Analysis (Insights)

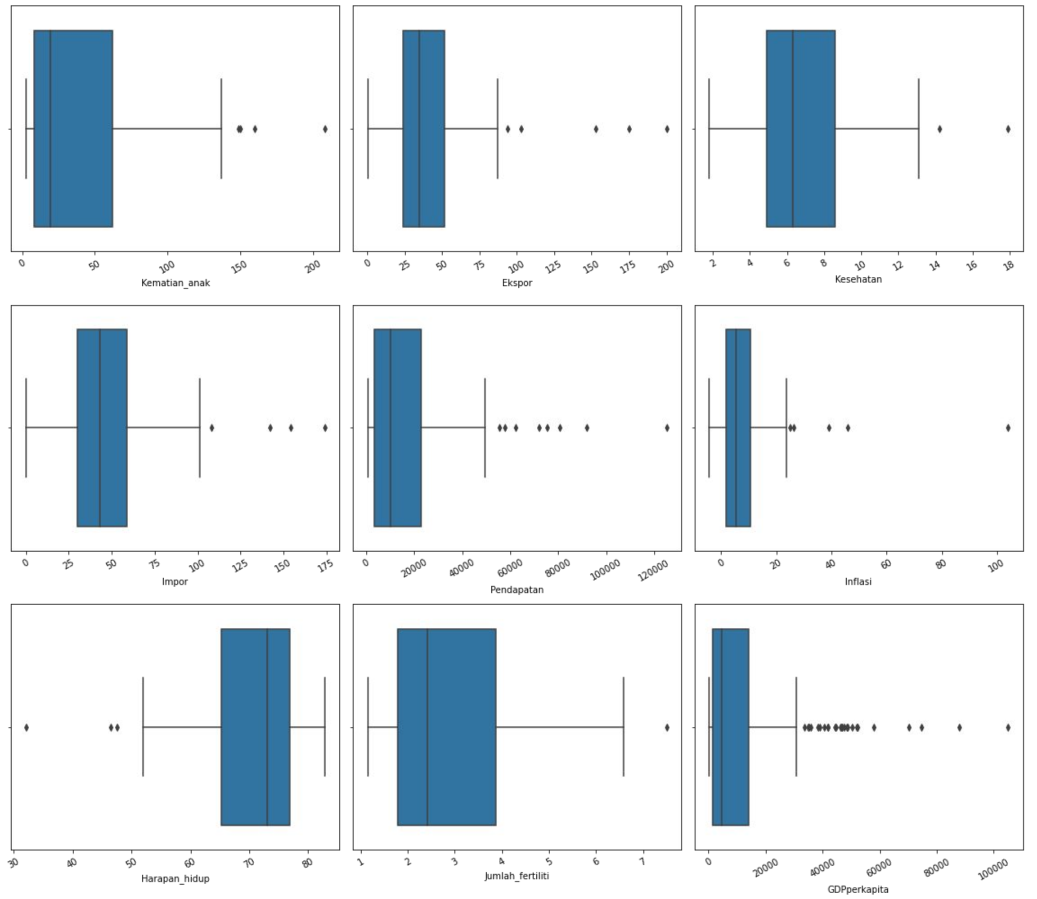
From pairplot and heatmap on previous slides, we can get several insights.

- 'GDPperkapita' and 'Pendapatan' have a high positive correlation (0.9). It means countries that have a high 'Pendapatan' will also have a high 'GDPperkapita'.
- 'Kematian_anak' and 'Jumlah_fertiliti' also have a high positive correlation (0.85).
- 'Impor' and 'Ekspor' also have a high positive correlation (0.74).
- 'Pendapatan' and 'Harapan_hidup' also have a high positive correlation (0.61).
- 'GDPperkapita' and 'Harapan_hidup' also have a high positive correlation (0.6).
- 'Harapan_hidup' and 'Jumlah_fertiliti' have a high negative correlation (-0.76). It means countries that have a high 'Harapan_hidup' will have a low 'Jumlah_fertiliti'.
- 'Harapan_hidup' and 'Kematian_anak' also have a high negative correlation (-0.89).

Outliers Treatment

03

Checking Outliers



Insights

From boxplots beside, we know that each columns of the dataframe have outliers. Although outliers can affect the results of the clustering, they cannot be removed. The removal of outliers will have an impact on the ranking of countries that need financial aid from HELP International. Hence, we will use another approach by capping the outliers because our objective is to find list of countries that need financial aid from HELP International. Therefore, we can cap a small part of the outliers. To minimalize bias, the capping will be based on 99th percentile.

The outliers are capped in these features (Most outliers features) : 'Ekspor', 'Impor' , 'Pendapatan', 'Kesehatan', 'Inflasi' and 'GDPperkapita'

The outliers are not capped in these features : 'Kematian_anak', 'Harapan_hidup' , and 'Jumlah_fertiliti'.

Handling Outliers

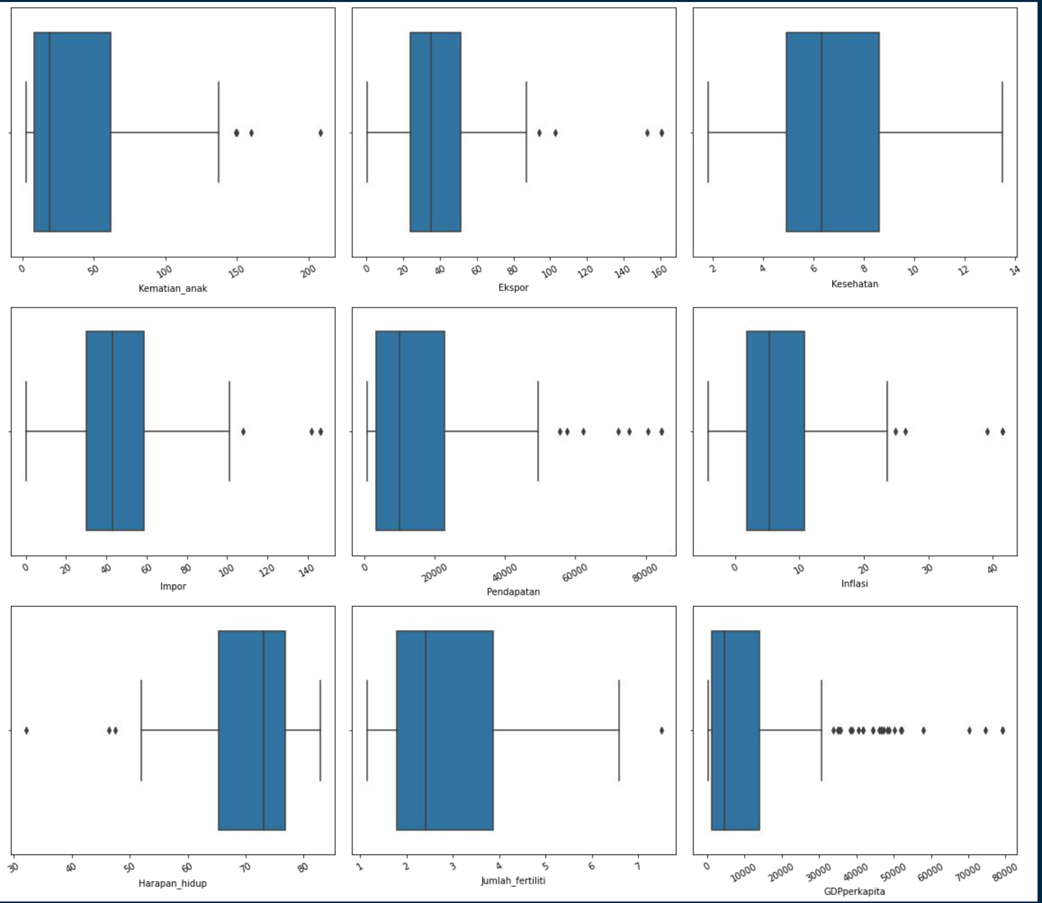
```
[ ]  # Cap the outliers
     cap_features = ['Ekspor', 'Kesehatan' ,'Impor' ,'Pendapatan' ,'Inflasi' , 'GDPperkapita']
     new_df = df.copy()
     for col in cap_features:
         q4 = new_df[col].quantile(0.99)
         new_df.loc[new_df[col] >= q4, col] = q4
```

! Informations !

There are different ranges in capping the outliers:

- Soft range: 1th and 99th percentile.
- Mid range: 5th and 95th percentile.
- 25th and 75th percentile.

After Handling Outliers



Insights

From boxplots beside, although columns of the dataframe still have outliers, we manage to minimalize a small part of outliers without causing large problem to the data

Scaling the Data

04

Dropping Non-Numeric Features

```
# Drop non-numeric column so we can rescale the data
num_df = new_df.drop(columns='Negara')
display(num_df)
```

	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita
0	90.2	10.0	7.58	44.9	1610.0	9.440	56.2	5.82	553.0
1	16.6	28.0	6.55	48.6	9930.0	4.490	76.3	1.65	4090.0
2	27.3	38.4	4.17	31.4	12900.0	16.100	76.5	2.89	4460.0
3	119.0	62.3	2.85	42.9	5900.0	22.400	60.1	6.16	3530.0
4	10.3	45.5	6.03	58.9	19100.0	1.440	76.8	2.13	12200.0
...	...	...	...	...	...	...	...	...	...
162	29.2	46.6	5.25	52.7	2950.0	2.620	63.0	3.50	2970.0
163	17.1	28.5	4.91	17.6	16500.0	41.478	75.4	2.47	13500.0
164	23.3	72.0	6.84	80.2	4490.0	12.100	73.1	1.95	1310.0
165	56.3	30.0	5.18	34.4	4480.0	23.600	67.5	4.67	1310.0
166	83.1	37.0	5.89	30.9	3280.0	14.000	52.0	5.40	1460.0

167 rows × 9 columns

! Informations !

Before we rescale the data, we must first drop every non-numeric features

Rescaling the Data

```
# Rescale the data using Standard Scaler
sc = StandardScaler()
scaled_df = sc.fit_transform(num_df)
scaled_df
```

```
array([[ 1.29153238, -1.19927911,  0.30123858, ..., -1.61909203,
         1.90288227, -0.70225949],
       [-0.5389489 , -0.49806893, -0.08896601, ...,  0.64786643,
        -0.85997281, -0.49872564],
       [-0.27283273, -0.09292528, -0.99060381, ...,  0.67042323,
        -0.0384044 , -0.47743428],
       ...,
       [-0.37231541,  1.21600038,  0.02089742, ...,  0.28695762,
        -0.66120626, -0.65869853],
       [ 0.44841668, -0.42015669, -0.60797601, ..., -0.34463279,
         1.14094382, -0.65869853],
       [ 1.11495062, -0.14746385, -0.33900002, ..., -2.09278484,
         1.6246091 , -0.6500669 ]])
```

```
# Check the top 5 rows of scaled dataframe
scaled_df = pd.DataFrame(scaled_df, columns = num_df.columns)
display(scaled_df.head())
```

	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita
0	1.291532	-1.199279	0.301239	-0.076771	-0.851668	0.265002	-1.619092	1.902882	-0.702259
1	-0.538949	-0.498069	-0.088966	0.083204	-0.386946	-0.372075	0.647866	-0.859973	-0.498726
2	-0.272833	-0.092925	-0.990604	-0.660465	-0.221053	1.122161	0.670423	-0.038404	-0.477434
3	2.007808	0.838126	-1.490672	-0.163244	-0.612045	1.932987	-1.179234	2.128151	-0.530950
4	-0.695634	0.183663	-0.285963	0.528541	0.125254	-0.764618	0.704258	-0.541946	-0.032042

! Informations !

To make the clustering more accurate, we standardize the data by rescaling it using the standard scaler provided by scikit-learn.

Clustering and their Visualization

05

Random 2-Clustering (Source Code & Insights)

```
# Clustering with n_cluster 2
kmeans1 = KMeans(n_clusters = 2, random_state = 42).fit(scaled_df)
labels1 = kmeans1.labels_
```

```
# Check after 2-clustering
print('n-cluster = 2 (Not a good cluster)')
print()
_tempdf = new_df.copy()
_tempdf['kmeans_2cluster'] = labels1
print('Cluster and its countries quantity :')
display(_tempdf.kmeans_2cluster.value_counts(ascending=True))
print()
display(_tempdf.head())
```

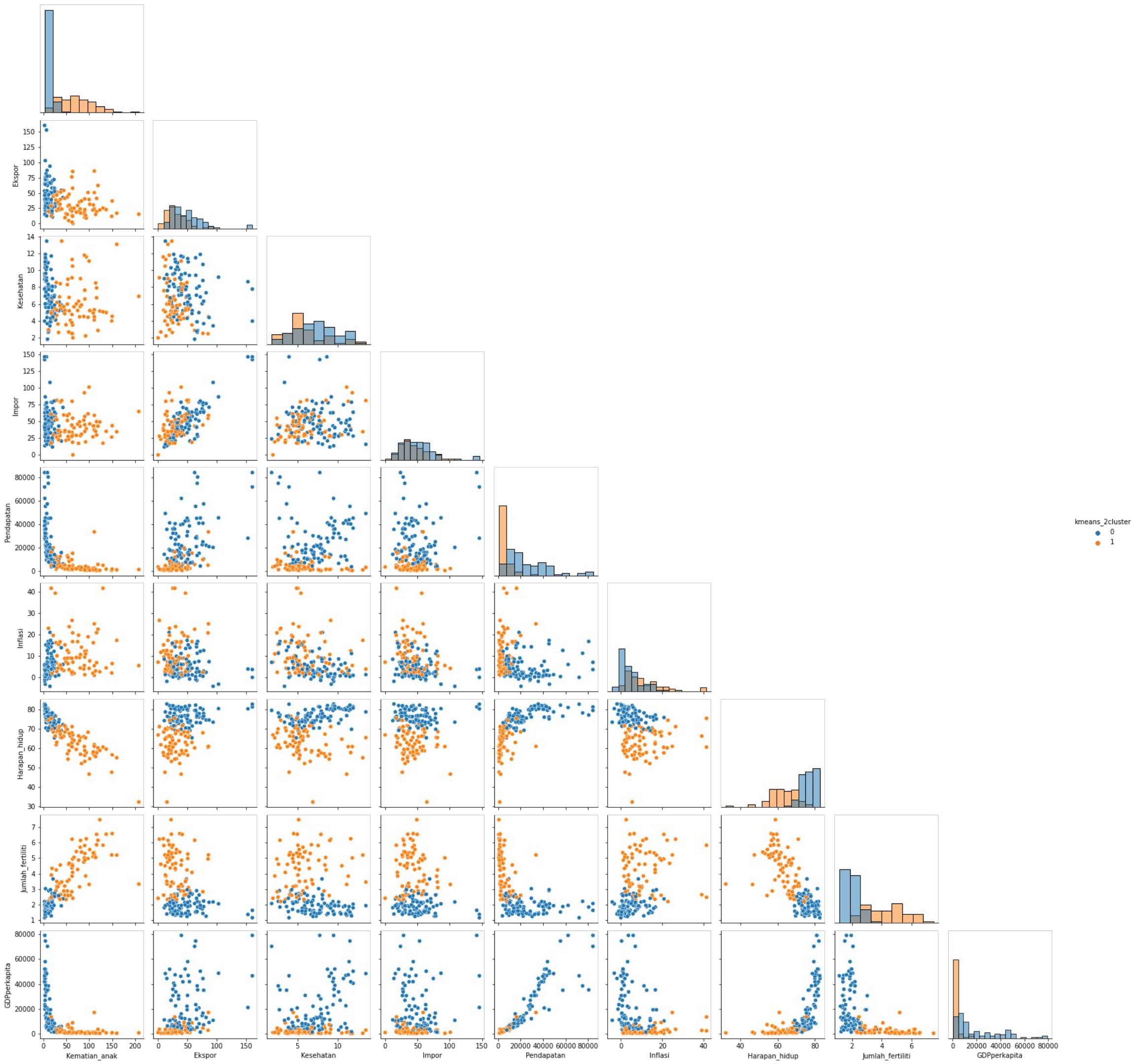
```
n-cluster = 2 (Not a good cluster)

Cluster and its countries quantity :
1    72
0    95
Name: kmeans_2cluster, dtype: int64
```

	Negara	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita	kmeans_2cluster
0	Afghanistan	90.2	10.0	7.58	44.9	1610.0	9.44	56.2	5.82	553.0	1
1	Albania	16.6	28.0	6.55	48.6	9930.0	4.49	76.3	1.65	4090.0	0
2	Algeria	27.3	38.4	4.17	31.4	12900.0	16.10	76.5	2.89	4460.0	1
3	Angola	119.0	62.3	2.85	42.9	5900.0	22.40	60.1	6.16	3530.0	1
4	Antigua and Barbuda	10.3	45.5	6.03	58.9	19100.0	1.44	76.8	2.13	12200.0	0

Insights

2-Clustering is not a good option and very few clusters. Therefore, we can use the elbow method or the silhouette score method to find other and more accurate cluster options.

Random 2-Clustering (Pair Plot)



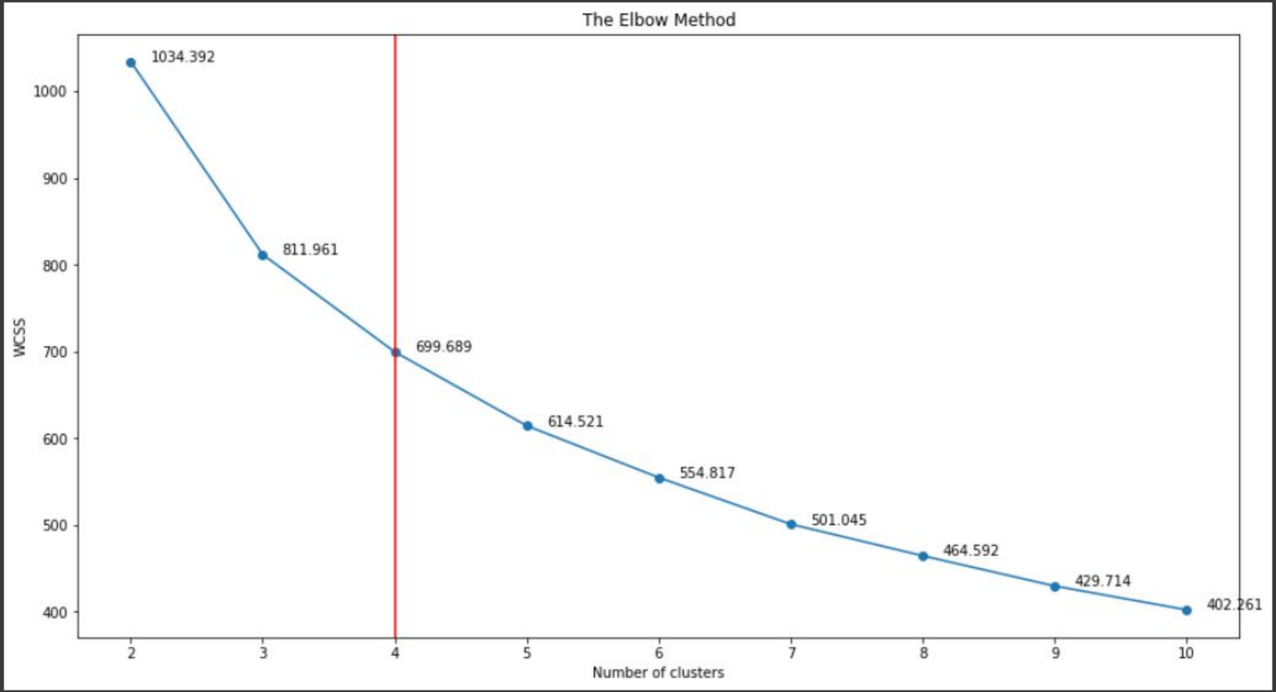
Elbow Method (Source Code)

```
# Elbow Method to find most accurate n-cluster
def elbowMethod(data, k_min=2, k_max= 10):
    wcss = [] # Within Cluster Sum of Squares
    k_range = range(k_min, k_max + 1)

    for i in k_range:
      kmeans_test = KMeans(n_clusters = i, random_state = 42, init = 'k-means++')
      kmeans_test.fit(data)
      wcss.append(kmeans_test.inertia_)

    fig, ax = plt.subplots(figsize=(15,8))
    ax.plot(k_range, wcss, marker='o')

    for i, value in enumerate(wcss):
        ax.text(i+2.15, value-0.005, round(value,3))

    plt.axvline(x = 4, color = 'r')
    plt.title('The Elbow Method')
    plt.xlabel('Number of clusters')
    plt.ylabel('WCSS')
    plt.show()
```

Elbow Method (Results)

```
print('Elbow Method')
print()
elbowMethod(scaled_df)
```

```
Elbow Method
```



Insights

We can see from elbow point, the best clusters we can get are 3-clusters or 4-clusters. Therefore, we should consider using silhouette method to make sure which clusters to used.

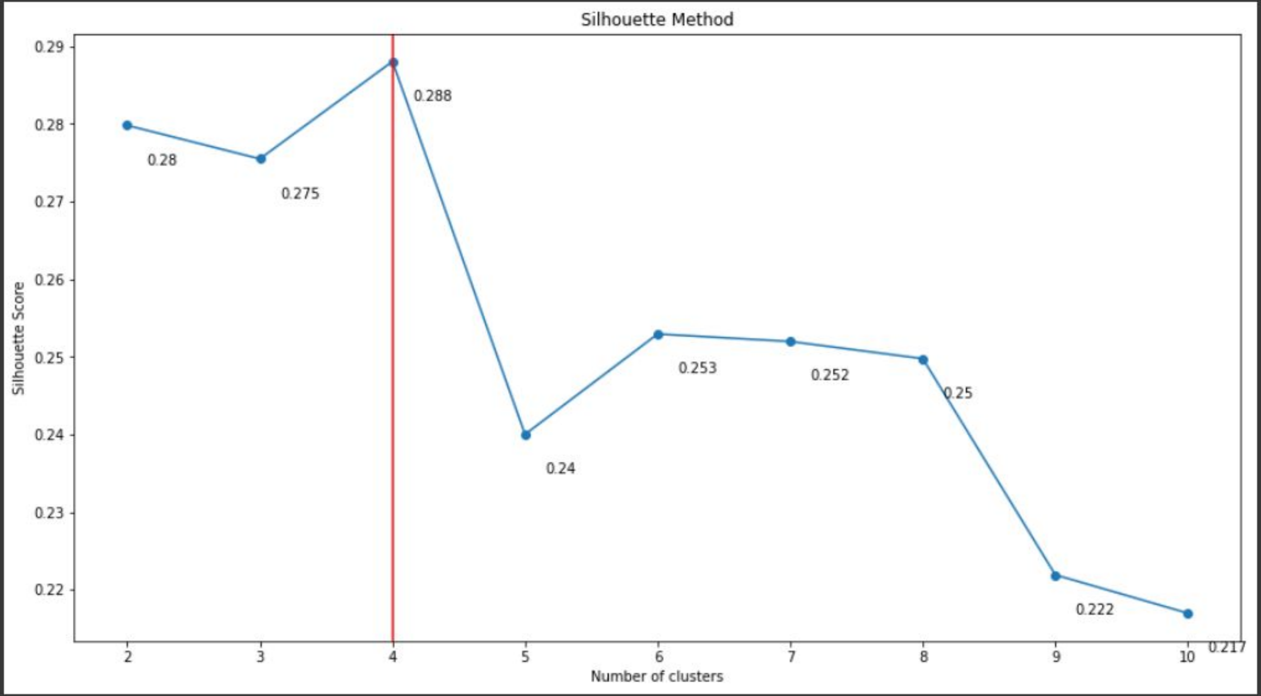
Silhouette Method (Source Code)

```python
# silhouette Method to find most accurate n-cluster
def silMethod(data, k_min=2, k_max=10):
    sil_score = []
    k_range = range(k_min, k_max+1)

    for k in k_range:
        model2 = KMeans(n_clusters = k)
        model2.fit(data)
        labels = model2.labels_
        s_score = silhouette_score(data, labels, metric='euclidean')
        sil_score.append(s_score)

    fig, ax = plt.subplots(figsize=(15,8))
    ax.plot(k_range, sil_score, marker='o')

    for i, value in enumerate(sil_score):
        ax.text(i+2.15, value-0.005, round(value,3))

    plt.xticks(k_range)
    plt.axvline(x = 4, color = 'r')
    plt.title('Silhouette Method')
    plt.xlabel('Number of clusters')
    plt.ylabel('Silhouette Score')
    plt.show()
```

Silhouette Method (Results)

```
print('Silhouette Method')
print()
silMethod(scaled_df)
```

```
Silhouette Method
```



Insights

We can see from silhouette score, the best cluster we can get is 4-clusters. Hence, we choose 4-clusters because they tend to have higher silhouette score.

4-Clustering (Source Code & Insights)

```
# Clustering with n_cluster 4
kmeans2 = KMeans(n_clusters = 4, random_state = 42).fit(scaled_df)
labels2 = kmeans2.labels_
```

```
# Check after 4-clustering
print('n-cluster = 4')
print()
new_df['Cluster'] = labels2
print('Cluster and its countries quantity :')
display(new_df.Cluster.value_counts(ascending=True))
print()
display(new_df.head())
```

```
n-cluster = 4

Cluster and its countries quantity :
3     5
1    31
2    46
0    85
Name: Cluster, dtype: int64
```

	Negara	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita	Cluster
0	Afghanistan	90.2	10.0	7.58	44.9	1610.0	9.44	56.2	5.82	553.0	2
1	Albania	16.6	28.0	6.55	48.6	9930.0	4.49	76.3	1.65	4090.0	0
2	Algeria	27.3	38.4	4.17	31.4	12900.0	16.10	76.5	2.89	4460.0	0
3	Angola	119.0	62.3	2.85	42.9	5900.0	22.40	60.1	6.16	3530.0	2
4	Antigua and Barbuda	10.3	45.5	6.03	58.9	19100.0	1.44	76.8	2.13	12200.0	0

Insights

4-Clustering is a good option and has enough clusters. Therefore, we will be using 4-clusters and we will do a full analysis in a later step.

4-Clustering (Mean Statistic Analysis)

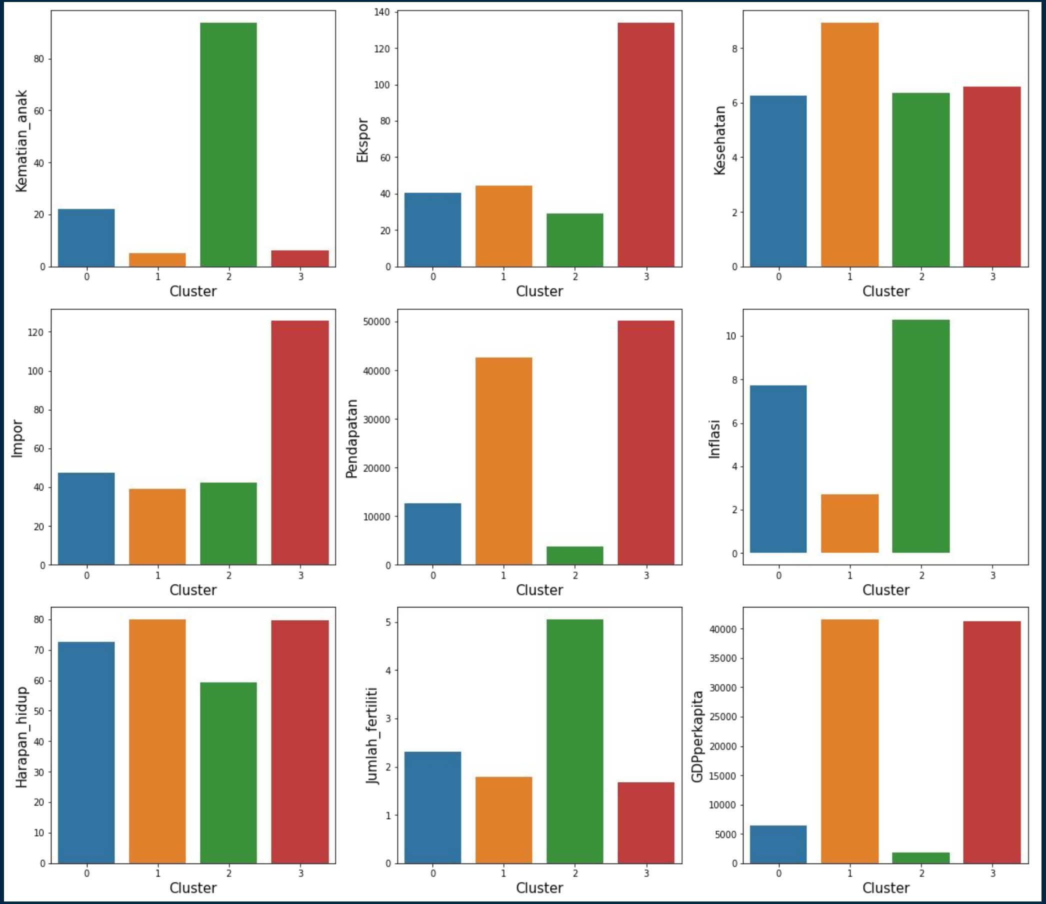
```
# Display mean statistic of each columns after 4-clustering to represents centers
analysis_res = new_df.groupby('Cluster').agg({'mean'})
analysis_res['Banyak_negara'] = new_df.groupby('Cluster')['Negara'].count()
display(analysis_res)
```

	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita	Banyak_negara
	mean	mean	mean	mean	mean	mean	mean	mean	mean	
Cluster										
0	22.138824	40.483400	6.246165	47.247834	12587.882353	7.711788	72.680000	2.309059	6467.200000	85
1	5.212903	44.283871	8.942387	39.103226	42657.225806	2.698806	80.070968	1.780000	41625.419355	31
2	93.841304	28.837174	6.346957	42.128261	3738.978261	10.727891	59.232609	5.054348	1826.130435	46
3	6.200000	134.152000	6.594000	125.732000	50174.800000	-0.005200	79.620000	1.672000	41257.600000	5

! Informations !

We will use mean of each columns after 4-clustering to determine which clusters represent underdeveloped countries, developing countries, developed countries, and well-developed countries.

4-Clustering (Multiple Bar Plots)



Insights

Based on the graphs beside, we should consider cluster 2 countries for aid recommendation because all of the data features representing cluster 2 are the closest to the characteristics of underdeveloped countries that need financial aid. Here are the reasons why we should choose cluster 2 as an option.

- Highest 'Kematian_anak'
- Lowest 'Ekspor'
- Comparatively low 'Kesehatan'
- Comparatively Low 'Impor'
- Lowest 'Pendapatan'
- Highest 'Inflasi'
- Lowest 'Harapan_hidup'
- Highest 'Jumlah_fertiliti'
- Lowest 'GDPperkapita'

Report Countries

06

Recommendation for HELP International

Due to financial limitations owned by HELP International, it is best to choose the most underdeveloped countries. Therefore, we will pick at least 5 of the most underdeveloped countries based by the following criterias.

- Highest 'Kematian_anak'
- Lowest 'Ekspor'
- Lowest 'Kesehatan'
- Highest 'Impor'
- Lowest 'Pendapatan'
- Highest 'Inflasi'
- Lowest 'Harapan_hidup'
- Highest 'Jumlah_fertiliti'
- Lowest 'GDPperkapita'

Results

```
# Show top 5 countries as a recommendation result
results = new_df[new_df['Cluster']==2]
results.sort_values(['GDPperkapita','Pendapatan','Kematian_anak','Kesehatan','Inflasi','Harapan_hidup','Jumlah_fertiliti','Impor','Ekspor'],
                    ascending=[True,True,False,True,False,True,False,False,True]).head()
```

	Negara	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita	Cluster
26	Burundi	93.6	8.92	11.60	39.2	764.0	12.30	57.7	6.26	231.0	2
88	Liberia	89.3	19.10	11.80	92.6	700.0	5.47	60.8	5.02	327.0	2
37	Congo, Dem. Rep.	116.0	41.10	7.91	49.6	609.0	20.80	57.5	6.54	334.0	2
112	Niger	123.0	22.20	5.16	49.1	814.0	2.55	58.8	7.49	348.0	2
132	Sierra Leone	160.0	16.80	13.10	34.5	1220.0	17.20	55.0	5.20	399.0	2

! Informations !

Showed from the code above, those are the top 5 countries recommended by KMeans Clustering

END OF SLIDES

Kelvin Erlangga
kelvinerlangga2002@gmail.com

THANKS