MILITARY DATA : CLASSIFICATION

Library req.

```
!pip install pyclustertend
!pip install termcolor
!pip install colorama
!pip install pyforest
!pip install ipython
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: pyclustertend in /usr/local/lib/python3.7/dist-packages (1.4.9)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: termcolor in /usr/local/lib/python3.7/dist-packages (1.1.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: colorama in /usr/local/lib/python3.7/dist-packages (0.4.5)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: pyforest in /usr/local/lib/python3.7/dist-packages (1.1.0)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: ipython in /usr/local/lib/python3.7/dist-packages (5.5.0)
Requirement already satisfied: decorator in /usr/local/lib/python3.7/dist-packages (from ipython) (4.4.2)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.7/dist-packages (from ipython) (0.7.5)
Requirement already satisfied: setuptools>=18.5 in /usr/local/lib/python3.7/dist-packages (from ipython) (57.4.0)
Requirement already satisfied: pexpect in /usr/local/lib/python3.7/dist-packages (from ipython) (4.8.0)
Requirement already satisfied: traitlets>=4.2 in /usr/local/lib/python3.7/dist-packages (from ipython) (5.1.1)
Requirement already satisfied: pygments in /usr/local/lib/python3.7/dist-packages (from ipython) (2.6.1)
Requirement already satisfied: simplegeneric>0.8 in /usr/local/lib/python3.7/dist-packages (from ipython) (0.8.1)
Requirement already satisfied: prompt-toolkit<2.0.0,>=1.0.4 in /usr/local/lib/python3.7/dist-packages (from ipython) (1.0.18)
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.7/dist-packages (from prompt-toolkit<2.0.0,>=1.0.4->ipython
Requirement already satisfied: wcwidth in /usr/local/lib/python3.7/dist-packages (from prompt-toolkit<2.0.0,>=1.0.4->ipython) (
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.7/dist-packages (from pexpect->ipython) (0.7.0)
```

```
import numpy as np
import pandas as pd
import seaborn as sns
import plotly.express as px
import matplotlib.pyplot as plt
import scipy.stats as stats
%matplotlib inline
import statsmodels.api as sm
import statsmodels.formula.api as smf

from sklearn.compose import make_column_transformer
from sklearn.cluster import KMeans
from pyclustertend import hopkins
from sklearn.metrics import silhouette_score
from yellowbrick.cluster import SilhouetteVisualizer
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import AgglomerativeClustering

from sklearn.preprocessing import scale
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import PolynomialFeatures
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import PowerTransformer
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import RobustScaler


import cufflinks as cf
import plotly.offline
cf.go_offline()
cf.set_config_file(offline=False, world_readable=True)

import warnings
warnings.filterwarnings('ignore')
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.filterwarnings("ignore", category=FutureWarning)
warnings.warn("this will not show")
```

```
plt.rcParams["figure.figsize"] = (10, 6)
pd.set_option('max_colwidth',200)
pd.set_option('display.max_rows', 1000)
pd.set_option('display.max_columns', 200)
pd.set_option('display.float_format', lambda x: '%.3f' % x)

from termcolor import colored
import missingno as msno

import colorama
from colorama import Fore, Style  # maakes strings colored
from termcolor import colored
```

```
def missing(df):
    missing_number = df.isnull().sum().sort_values(ascending=False)
    missing_percent = (df.isnull().sum()/df.isnull().count()).sort_values(ascending=False)
    missing_values = pd.concat([missing_number, missing_percent], axis=1, keys=['Missing_Number', 'Missing_Percent'])
    return missing_values

def missing_values(df):
    return missing(df)[missing(df)['Missing_Number']>0]

def first_looking(df):
    print(colored("Shape:", attrs=['bold']), df.shape,'\n',
          f"There is ", df.shape[0], " observation and ", df.shape[1], " columns in the dataset.", '\n',
          colored('-'*79, 'red', attrs=['bold']),
          colored("\nInfo:\n", attrs=['bold']), sep='')
    print(df.info(), '\n',
          colored('-'*79, 'red', attrs=['bold']), sep='')
    print(colored("Number of Uniques:\n", attrs=['bold']), df.nunique(),'\n',
          colored('-'*79, 'red', attrs=['bold']), sep='')
    print(colored("Missing Values:\n", attrs=['bold']), missing_values(df),'\n',
          colored('-'*79, 'red', attrs=['bold']), sep='')
```

```
    print(colored("All Columns:", attrs=['bold']), list(df.columns),'\n',
          colored('-'*79, 'red', attrs=['bold']), sep='')

    df.columns= df.columns.str.lower().str.replace('&', '_').str.replace(' ', '_')

    print(colored("Columns after rename:", attrs=['bold']), list(df.columns),'\n',
              colored('-'*79, 'red', attrs=['bold']), sep='')

def duplicate_values(df):
    duplicate_values = df.duplicated(subset=None, keep='first').sum()
    if duplicate_values > 0:
        df.drop_duplicates(keep='first', inplace=True)
        print(duplicate_values, colored("duplicates were dropped", attrs=['bold']),'\n',
              colored('-'*79, 'red', attrs=['bold']), sep='')
    else:
        print(colored("No duplicates", attrs=['bold']),'\n',
              colored('-'*79, 'red', attrs=['bold']), sep='')

def drop_columns(df, drop_columns):
    if drop_columns !=[]:
        df.drop(drop_columns, axis=1, inplace=True)
        print(drop_columns, 'were dropped')
    else:
        print(colored('We will now check the missing values and if necessary drop some columns!!!', attrs=['bold']),'\n',
              colored('-'*79, 'red', attrs=['bold']), sep='')

def drop_null(df, limit):
    print('Shape:', df.shape)
    for i in df.isnull().sum().index:
        if (df.isnull().sum()[i]/df.shape[0]*100)>limit:
            print(df.isnull().sum()[i], 'percent of', i ,'null and were dropped')
            df.drop(i, axis=1, inplace=True)
            print('new shape:', df.shape)
        else:
            print(df.isnull().sum()[i], '%, percentage of missing values of', i ,'less than limit', limit, '%, so we will keep it.')
    print('New shape after missing value control:', df.shape)
```

```
def missing (df):
    missing_number = df.isnull().sum().sort_values(ascending=False)
    missing_percent = (df.isnull().sum()/df.isnull().count()).sort_values(ascending=False)
    missing_values = pd.concat([missing_number, missing_percent], axis=1, keys=['Missing_Number', 'Missing_Percent'])
    return missing_values
```

```
def first_look(col):
    print("column name    : ", col)
    print("--------------------------------")
    print("per_of_nulls   : ", "%", round(df[col].isnull().sum()/df.shape[0]*100, 2))
    print("num_of_nulls   : ", df[col].isnull().sum())
    print("num_of_uniques : ", df[col].nunique())
    print(df[col].value_counts(dropna = False))
```

Load Dataset

```
# from local drive
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving World military power.csv to World military power.csv
```

```
df0 = pd.read_csv("World military power.csv", header=1)
df = df0.copy()
```

```
df.head()
```

	Military Strength	Military Strength Power Index	Aircraft Strength	Aircraft Strength value	Fighter/Interceptor Strength	Fighter/Interceptor Strength value	Attack Aircraft Strength	A Air Str
0	Afghanistan	1.344	Afghanistan	260	Afghanistan	0	Afghanistan	:
1	Albania	2.314	Albania	19	Albania	0	Albania	
2	Algeria	0.466	Algeria	551	Algeria	103	Algeria	:
3	Angola	0.838	Angola	295	Angola	72	Angola	'
4	Argentina	0.652	Argentina	227	Argentina	24	Argentina	

Dataset Details

```
df.columns
```

```
Index(['military_strength', 'military_strength_power_index',
       'aircraft_strength', 'aircraft_strength_value',
       'fighter/interceptor_strength', 'fighter/interceptor_strength_value',
       'attack_aircraft_strength', 'attack_aircraft_strength_value',
       'transport_aircraft_fleet_strength',
       'transport_aircraft_fleet_strength_value', 'trainer_aircraft_fleet',
       'trainer_aircraft_fleet_value', 'helicopter_fleet_strength',
       'helicopter_fleet_strength_value', 'attack_helicopter_fleet_strength',
       'attack_helicopter_fleet_strength_value', 'tank_strength',
       'tank_strength_value', 'afv/apc_strength', 'afv/apc_strength_value',
       'self-propelled_artillery_strength',
       'self-propelled_artillery_strength_value', 'towed_artillery_strength',
       'towed_artillery_strength_value', 'rocket_projector_strength',
       'rocket_projector_strength_value', 'navy_fleet_strengths',
       'navy_fleet_strengths_value', 'aircraft_carrier_fleet_strength',
       'aircraft_carrier_fleet_strength_value', 'submarine_fleet_strength',
       'submarine_fleet_strength_value', 'destroyer_fleet_strength',
```

```
       'destroyer_fleet_strength_value', 'frigate_fleet_strength',
       'frigate_fleet_strength_value', 'defense_spending_budget',
       'defense_spending_budget_value', 'external_debt', 'external_debt_value',
       'airport_totals', 'airport_totals_value', 'oil_production',
       'oil_production_value', 'oil_consumption', 'oil_consumption_value',
       'proven_oil_reserves', 'proven_oil_reserves_value',
       'available_manpower', 'available_manpower_value', 'total_population',
       'total_population_value', 'total_square_land_area',
       'total_square_land_area_value', 'total_coastline_coverage',
       'total_coastline_coverage_value', 'total_waterway_coverage',
       'total_waterway_coverage_value', 'total_border_coverage',
       'total_border_coverage_value'],
      dtype='object')
```

DATASET SUMMARY

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
military_strength_power_index	138.000	1.461	1.324	0.061	0.575	1.034	2.022	10.168
attack_aircraft_strength_value	138.000	25.761	94.528	0.000	0.000	0.000	15.750	742.000
transport_aircraft_fleet_strength_value	138.000	30.232	92.330	0.000	3.000	9.000	26.000	945.000
attack_helicopter_fleet_strength_value	138.000	25.623	97.326	0.000	0.000	2.000	17.750	967.000
navy_fleet_strengths_value	124.000	84.984	146.114	0.000	10.000	38.000	77.750	984.000
aircraft_carrier_fleet_strength_value	115.000	0.383	1.972	0.000	0.000	0.000	0.000	20.000
submarine_fleet_strength_value	115.000	4.800	13.707	0.000	0.000	0.000	4.000	83.000
destroyer_fleet_strength_value	115.000	2.052	10.001	0.000	0.000	0.000	0.000	91.000
frigate_fleet_strength_value	115.000	3.522	6.481	0.000	0.000	0.000	5.500	52.000

Handling Duplicate value

```
df.duplicated().value_counts()
```

```
False    139
True      26
dtype: int64
```

```
duplicate
```

	military_strength	military_strength_power_index	aircraft_strength	aircraft_strength_value	fig
139	NaN	NaN	NaN	NaN	
140	NaN	NaN	NaN	NaN	
141	NaN	NaN	NaN	NaN	
142	NaN	NaN	NaN	NaN	
143	NaN	NaN	NaN	NaN	
144	NaN	NaN	NaN	NaN	
145	NaN	NaN	NaN	NaN	
146	NaN	NaN	NaN	NaN	
147	NaN	NaN	NaN	NaN	
148	NaN	NaN	NaN	NaN	
149	NaN	NaN	NaN	NaN	
150	NaN	NaN	NaN	NaN	
151	NaN	NaN	NaN	NaN	

Now Remove Multiple Countary Name

153	NaN	NaN	NaN	NaN

```
df.rename(columns={'military_strength': 'country'}, inplace=True)
```

```
df.head(1)
```

	country	military_strength_power_index	aircraft_strength	aircraft_strength_value	fighter/inter

```
df.head()
```

	country	military_strength_power_index	aircraft_strength	aircraft_strength_value	fighter/inter
0	Afghanistan	1.344	Afghanistan	260	
1	Albania	2.314	Albania	19	
2	Algeria	0.466	Algeria	551	
3	Angola	0.838	Angola	295	
4	Argentina	0.652	Argentina	227	

Handling Missing Data

```
missing(df)
```

	Missing_Number	Missing_Percent
total_coastline_coverage_value	29	0.210
total_coastline_coverage	29	0.210
submarine_fleet_strength	23	0.167
frigate_fleet_strength_value	23	0.167
frigate_fleet_strength	23	0.167
destroyer_fleet_strength_value	23	0.167
destroyer_fleet_strength	23	0.167
submarine_fleet_strength_value	23	0.167
aircraft_carrier_fleet_strength_value	23	0.167
aircraft_carrier_fleet_strength	23	0.167
navy_fleet_strengths	14	0.101
navy_fleet_strengths_value	14	0.101
total_border_coverage	9	0.065
total_border_coverage_value	9	0.065
attack_helicopter_fleet_strength	0	0.000
available_manpower	0	0.000
external_debt_value	0	0.000
airport_totals	0	0.000
airport_totals_value	0	0.000
oil_production	0	0.000
oil_production_value	0	0.000
oil_consumption	0	0.000

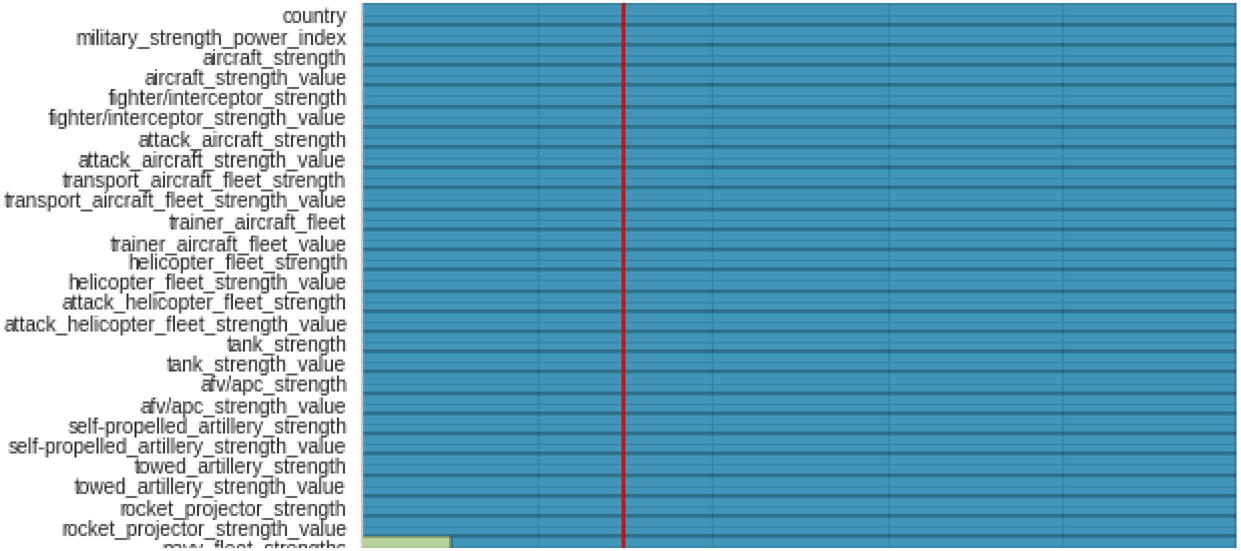
oil_consumption_value	0	0.000
proven_oil_reserves	0	0.000
proven_oil_reserves_value	0	0.000

```
plt.figure(figsize=(4,6))

sns.displot(
    data=df.isnull().melt(value_name="missing"),
    y="variable",
    hue="missing",
    multiple="fill",
    height=9.25
)

plt.axvline(0.3,color="r");
```

<Figure size 288x432 with 0 Axes>




```
plt.figure(figsize=(4,6))

sns.displot(
    data=value_df.isnull().melt(value_name="missing"),
    y="variable",
    hue="missing",
    multiple="fill",
    height=9.25
)

plt.axvline(0.3,color="r");
```

```
<Figure size 288x432 with 0 Axes>
```


```
value_df = value_df.reset_index()
del value_df['index']
```

```
for col in value_df.columns:
    if value_df.dtypes[col] == np.object:
        value_df[col] = value_df[col].apply(lambda x : x.replace(",", "") if type(x) != int else x).astype("float")
```

```
value_df.head()
```

country	Military Strength Power Index	aircraft_strength_value	fighter/interceptor_strength_value	attack_aircraft_str
Afghanistan	1.344	260.000	0.000	
Albania	2.314	19.000	0.000	
Algeria	0.466	551.000	103.000	
Angola	0.838	295.000	72.000	
Argentina	0.652	227.000	24.000	

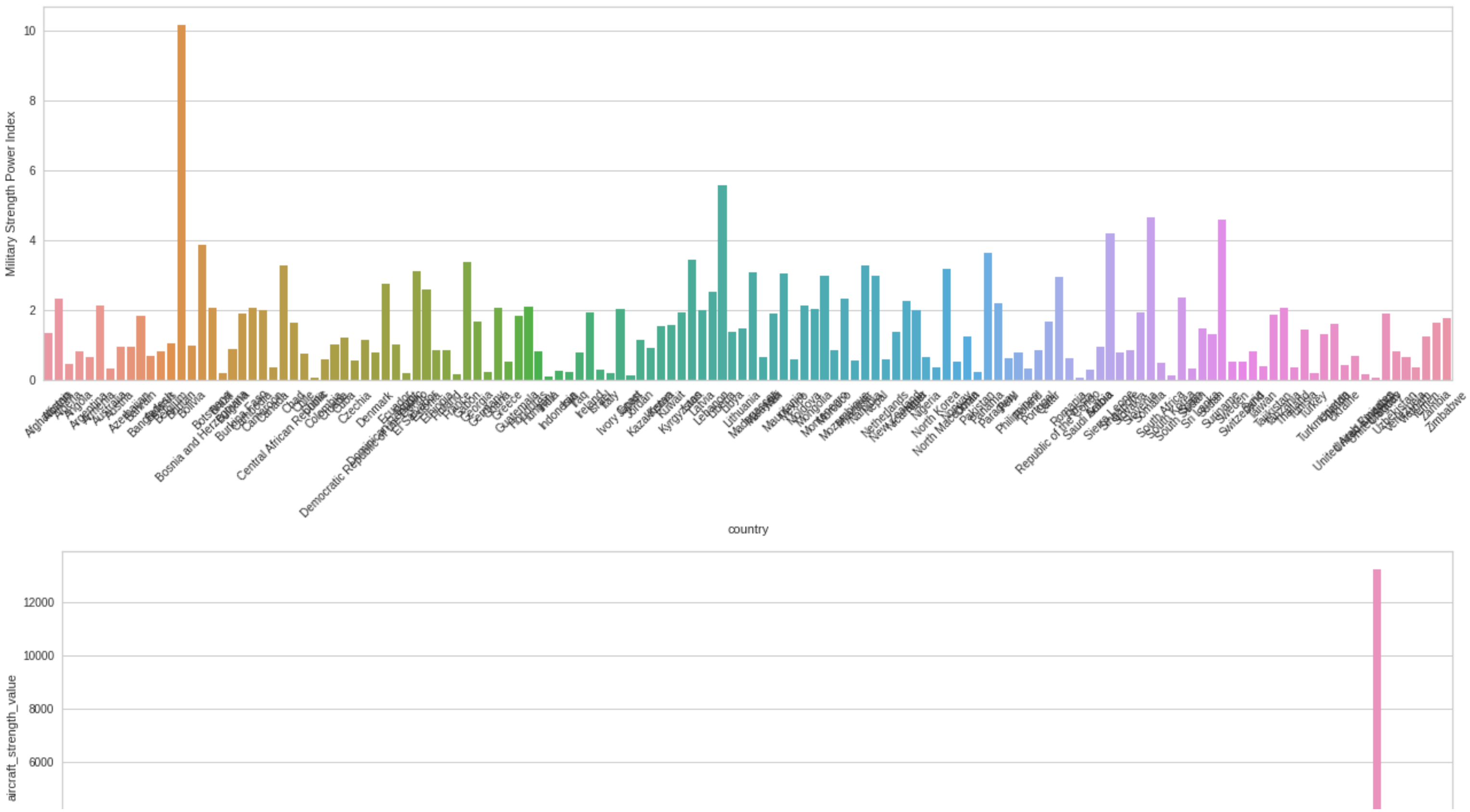
All Missing value corrected

```
missing(value_df)
```

	Missing_Number	Missing_Percent
Military Strength Power Index	0	0.000
aircraft_strength_value	0	0.000
total_waterway_coverage_value	0	0.000
total_coastline_coverage_value	0	0.000
total_square_land_area_value	0	0.000
total_population_value	0	0.000
available_manpower_value	0	0.000
proven_oil_reserves_value	0	0.000
oil_consumption_value	0	0.000
oil_production_value	0	0.000
airport_totals_value	0	0.000
external_debt_value	0	0.000
defense_spending_budget_value	0	0.000
frigate_fleet_strength_value	0	0.000
destroyer_fleet_strength_value	0	0.000
submarine_fleet_strength_value	0	0.000
aircraft_carrier_fleet_strength_value	0	0.000
navy_fleet_strengths_value	0	0.000
rocket_projector_strength_value	0	0.000
towed_artillery_strength_value	0	0.000
self-propelled_artillery_strength_value	0	0.000
afv/apc_strength_value	0	0.000

tank_strength_value	0	0.000

```
for col in value_df.columns:
    plt.figure(figsize = (22,6))
    sns.barplot(y = value_df[col], x = value_df.index, data = value_df)
    plt.xticks(rotation = 45);
```




```
value_df.describe().T.style.background_gradient(subset=['mean','std','50%','count'], cmap='RdPu')
```

	count	mean	std	
Military Strength Power Index	138.000000	1.460716	1.324018	0.06(
aircraft_strength_value	138.000000	388.471014	1231.981859	0.00(
fighter/interceptor_strength_value	138.000000	81.565217	230.324777	0.00(
attack_aircraft_strength_value	138.000000	25.760870	94.528222	0.00(
transport_aircraft_fleet_strength_value	138.000000	30.231884	92.330436	0.00(
trainer_aircraft_fleet_value	138.000000	82.833333	240.803721	0.00(
helicopter_fleet_strength_value	138.000000	154.065217	520.183631	0.00(
attack_helicopter_fleet_strength_value	138.000000	25.623188	97.326091	0.00(
tank_strength_value	138.000000	646.565217	1515.463683	0.00(
afv/apc_strength_value	138.000000	2485.695652	5410.546392	0.00(
self-propelled_artillery_strength_value	138.000000	212.159420	688.028987	0.00(
towed_artillery_strength_value	138.000000	393.978261	805.781703	0.00(
rocket_projector_strength_value	138.000000	156.934783	484.380984	0.00(
navy_fleet_strengths_value	138.000000	74.362319	141.038222	0.00(
aircraft_carrier_fleet_strength_value	138.000000	0.318841	1.804235	0.00(
submarine_fleet_strength_value	138.000000	3.978261	12.603132	0.00(
destroyer_fleet_strength_value	138.000000	1.710145	9.154954	0.00(
frigate_fleet_strength_value	138.000000	2.891304	5.969123	0.00(
defense_spending_budget_value	138.000000	13993631641.210144	67311892724.125046	13000000.00(
external_debt_value	138.000000	519498263043.478271	1847251527661.875244	539400000.00(
oil_production_value	138.000000	585552.355072	1667138.897000	0.00(

Data Preprocessing

```
value_df.head()
```

country	Military Strength Power Index	aircraft_strength_value	fighter/interceptor_strength_value	attack_aircraft_str
Afghanistan	1.344	260.000	0.000	
Albania	2.314	19.000	0.000	
Algeria	0.466	551.000	103.000	
Angola	0.838	295.000	72.000	
Argentina	0.652	227.000	24.000	


```
value_df.reset_index(inplace=True)
```

```
value_df.head()
```

	country	Military Strength Power Index	aircraft_strength_value	fighter/interceptor_strength_value	attack_aircraft_s
0	Afghanistan	1.344	260.000	0.000	
2	Algeria	0.466	551.000	103.000	
4	Argentina	0.652	227.000	24.000	

Scaling Model

```
X = value_df.drop(["country"], axis=1)
```

```
scaler = scale
X_scaled = pd.DataFrame(scaler(X))
X_scaled.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	
0	-0.088	-0.105	-0.355	-0.008	-0.003	-0.345	0.064	-0.264	-0.428	-0.264	-0.309	-0.272	-0.222	-0.529	-
1	0.647	-0.301	-0.355	-0.274	-0.329	-0.345	-0.261	-0.264	-0.428	-0.374	-0.309	-0.491	-0.325	-0.259	-
2	-0.754	0.132	0.093	-0.040	0.313	0.017	0.199	0.200	0.155	0.904	0.157	-0.192	0.330	0.901	-
3	-0.472	-0.076	-0.042	-0.082	-0.003	-0.149	-0.054	-0.110	-0.177	-0.351	-0.269	-0.046	-0.087	-0.124	-
4	-0.613	-0.132	-0.251	-0.199	-0.231	-0.078	-0.104	-0.264	-0.183	-0.324	-0.188	-0.276	-0.271	-0.180	-


```
X_scaled.shape
```

```
(138, 26)
```

MODELLING WITH K-MEAN

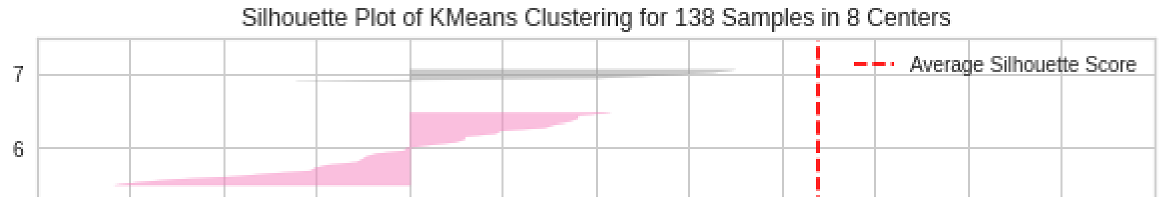
```
K_means_model1 = KMeans(random_state = 101)
```

```
K_means_model1.fit_predict(X)
```

```
array([0, 0, 0, 0, 0, 0, 3, 6, 0, 0, 0, 0, 3, 0, 0, 0, 0, 6, 0, 0, 0, 0,
       3, 0, 0, 0, 3, 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 2,
       0, 6, 0, 0, 0, 6, 6, 0, 0, 7, 0, 7, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0,
       0, 6, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 6, 0, 0, 0, 6, 0, 0, 0, 0,
       6, 0, 7, 0, 0, 0, 6, 3, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 4, 1, 0,
       0, 0, 0, 0, 0, 0], dtype=int32)
```

```
K_means_model1 = KMeans(n_clusters=8, random_state=101)
visualizer = SilhouetteVisualizer(K_means_model1)

visualizer.fit(X_scaled)    # Fit the data to the visualizer
visualizer.poof();
```




```
# With optimal K value

K_means_model2 = KMeans(n_clusters = 3, random_state = 101)

K_means_model2.fit_predict(X_scaled)
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0], dtype=int32)
```

−0.4 −0.3 −0.2 −0.1 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8

```
K_means_model2.labels_
```

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0], dtype=int32)
```

```
K_means_model2.inertia_
```

```
1306.8560688687046
```

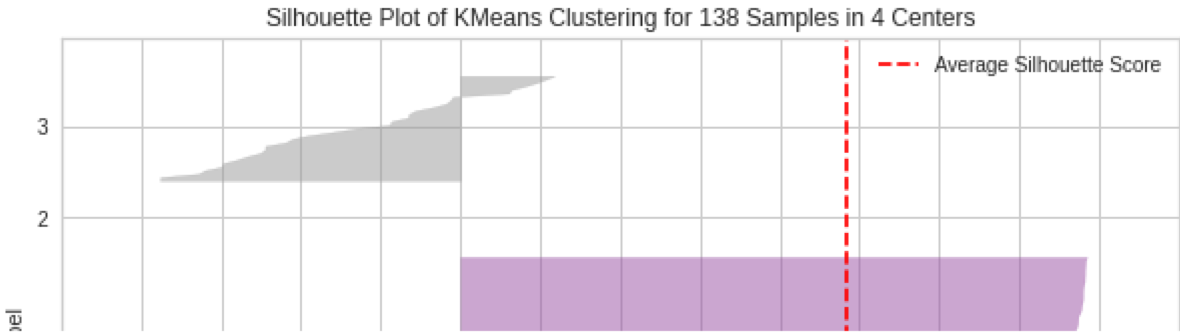
```
value_df["K-Means_cluster"] = K_means_model2.labels_
```

```
value_df.head()
```

	country	Military Strength Power Index	aircraft_strength_value	fighter/interceptor_strength_value	attack_aircraft_s
0	Afghanistan	1.344	260.000	0.000	
1	Albania	2.314	19.000	0.000	
2	Algeria	0.466	551.000	103.000	
3	Angola	0.838	295.000	72.000	
4	Argentina	0.652	227.000	24.000	

```
K_means_model2 = KMeans(n_clusters=4, random_state=101)
visualizer = SilhouetteVisualizer(K_means_model2)

visualizer.fit(X_scaled)    # Fit the data to the visualizer
visualizer.poof();
```




```
value_df["K-Means_cluster"].value_counts()
```

```
0    133
2      4
1      1
Name: K-Means_cluster, dtype: int64
```

Country Ranking

silhouette coefficient values

```
value_df[value_df["K-Means_cluster"] == 0][["country", "Military Strength Power Index"]].sort_values(by="Military Strength Power Inde
```

	country	Military Strength Power Index
57	Japan	0.150
110	South Korea	0.151
40	France	0.170
129	United Kingdom	0.172
35	Egypt	0.187
17	Brazil	0.199
124	Turkey	0.210
55	Italy	0.211
43	Germany	0.219
51	Iran	0.219
91	Pakistan	0.236
50	Indonesia	0.254
102	Saudi Arabia	0.303
54	Israel	0.311
6	Australia	0.323
112	Spain	0.339
96	Poland	0.340
134	Vietnam	0.356
122	Thailand	0.357
22	Canada	0.371
119	Taiwan	0.401
127	Ukraine	0.446

```
value_df[value_df["K-Means_cluster"] == 1][["country", "Military Strength Power Index"]].sort_values(by="Military Strength Power Inde
```

	country	Military Strength Power Index
130	United States	0.061
116	Sweden	0.530

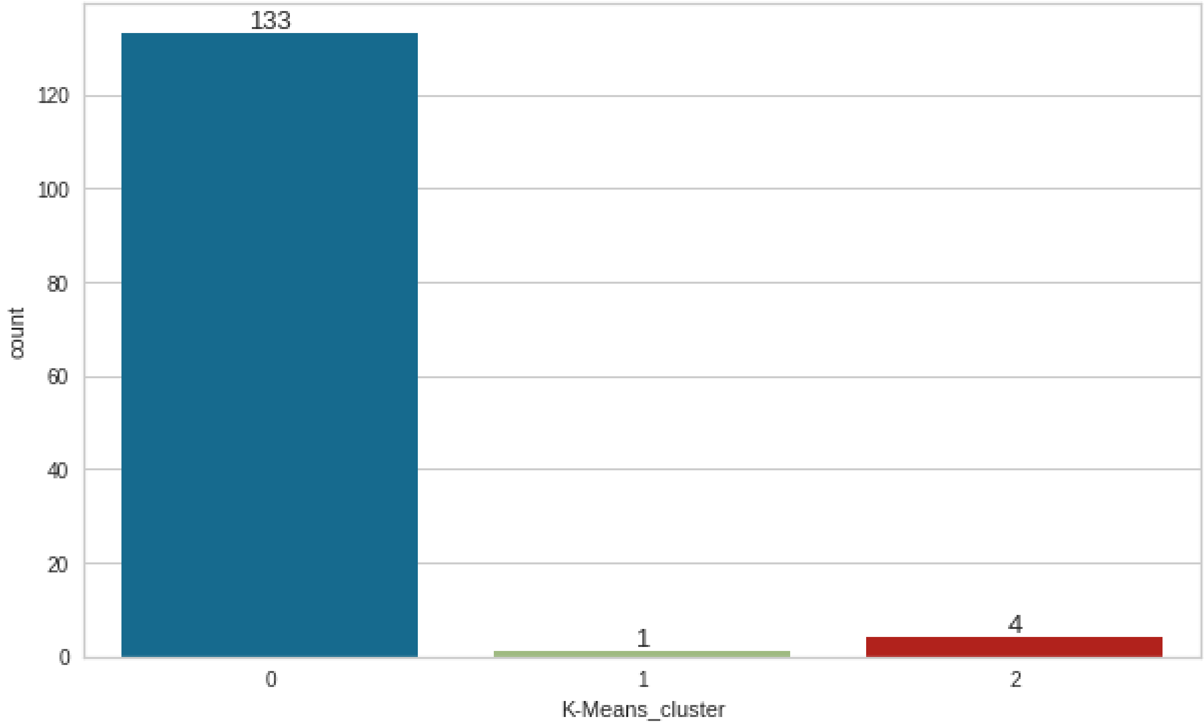
```
value_df[value_df["K-Means_cluster"] == 2][["country", "Military Strength Power Index"]].sort_values(by="Military Strength Power Inde
```

	country	Military Strength Power Index
101	Russia	0.068
26	China	0.069
49	India	0.095
87	North Korea	0.372

```
value_df["K-Means_cluster"].value_counts()
```

```
sns.countplot(x=value_df["K-Means_cluster"], data=value_df)

for index, value in enumerate(value_df["K-Means_cluster"].value_counts().sort_index()):
    plt.text(index, value, f"{value}", ha="center", va="bottom", fontsize = 13)
```



59	Kazakhstan	0.910
8	Azerbaijan	0.946
103	Serbia	0.948
7	Austria	0.957
14	Bolivia	0.994
34	Ecuador	1.006
28	Croatia	1.018