0.) Import and Clean data

In [ ]:
```
import pandas as pd
# from google.colab import drive
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
```

In [ ]:
```
#drive.mount('/content/gdrive/', force_remount = True)
df = pd.read_csv("Country-data.csv", sep = ",")
```

In [ ]:
```
df.head()
```

Out[ ]:

	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

In [ ]:
```
names = df[['country']].copy()
X = df.drop('country',axis=1)
```

In [ ]:
```
scale = StandardScaler().fit(X)
X_scaled = scale.transform(X)
```

1.) Fit a kmeans Model with any Number of Clusters

In [ ]:
```
kmeans = KMeans(n_clusters = 5)
kmeans.fit(X_scaled)
```

```
/Users/laoga/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:
The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` expl
icitly to suppress the warning
  warnings.warn(
```

Out[ ]:

```
▾ KMeans
KMeans(n_clusters=5)
```

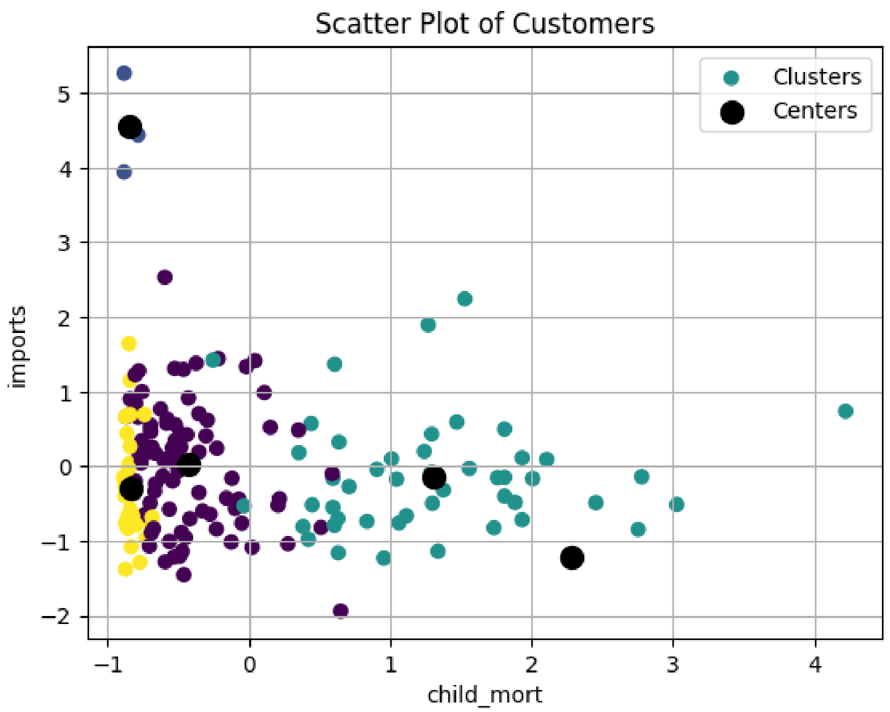
2.) Pick two features to visualize across

In [ ]:
```
X.columns
```

Out[ ]:
```
Index(['child_mort', 'exports', 'health', 'imports', 'income', 'inflation',
       'life_expec', 'total_fer', 'gdpp'],
      dtype='object')
```

In [ ]:
```
import matplotlib.pyplot as plt

x1_index = 0
x2_index = 3


scatter = plt.scatter(X_scaled[:, x1_index], X_scaled[:, x2_index], c=kmeans.labels_, cmap='viridis


centers = plt.scatter(kmeans.cluster_centers_[:, x1_index], kmeans.cluster_centers_[:, x2_index], r

plt.xlabel(X.columns[x1_index])
plt.ylabel(X.columns[x2_index])
plt.title('Scatter Plot of Customers')

# Generate legend
plt.legend()
```

```
plt.grid()
plt.show()
```



3.) Check a range of k-clusters and visualize to find the elbow. Test 30 different random starting places for the centroid means

In [ ]:
```
WCSSs = []
Ks = range(1,15)
for k in Ks:
    kmeans = KMeans(n_clusters = k, n_init=30).fit(X_scaled)
    WCSSs.append(kmeans.inertia_)
```

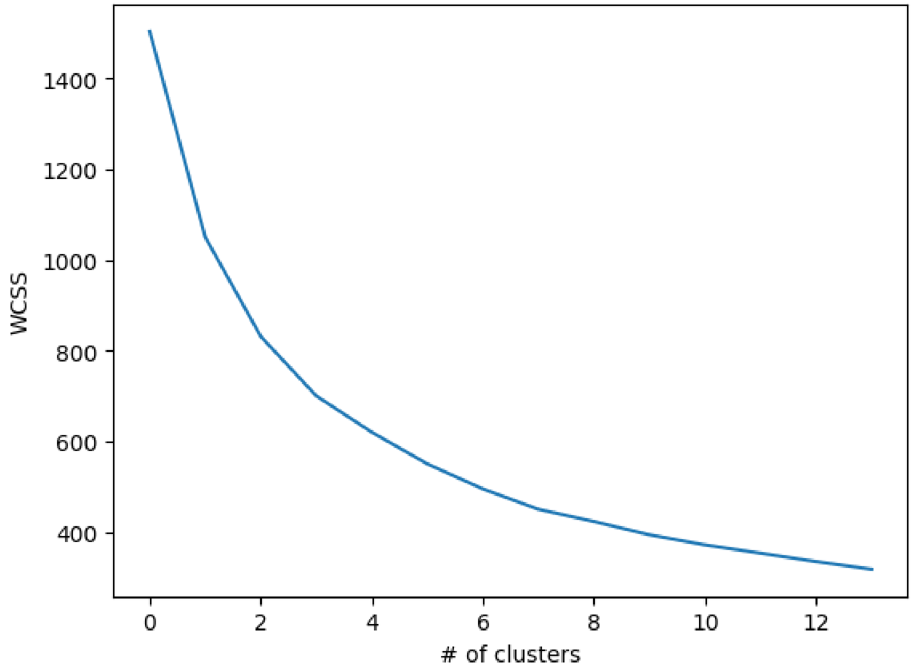
In [ ]:
```
# Optional: do in 1 line of code
WCSSs = [KMeans(n_clusters = k, n_init=30).fit(X_scaled).inertia_ for k in range(1,15)]
```

In [ ]:
```
WCSSs
```

Out[ ]:
```
[1503.0,
 1050.2145582853304,
 831.4244352086874,
 700.3229986404374,
 620.3621532663786,
 550.5699592955896,
 495.3233825951919,
 450.53083287148144,
 423.5717587733913,
 394.2738710166505,
 372.03682472739024,
 353.7761205673606,
 335.34849629584994,
 318.31967154943004]
```

4.) Use the above work and economic critical thinking to choose a number of clusters. Explain why you chose the number of clusters and fit a model accordingly.

In [ ]:
```
plt.plot(WCSSs)
plt.xlabel('# of clusters')
plt.ylabel('WCSS')
plt.show()
```

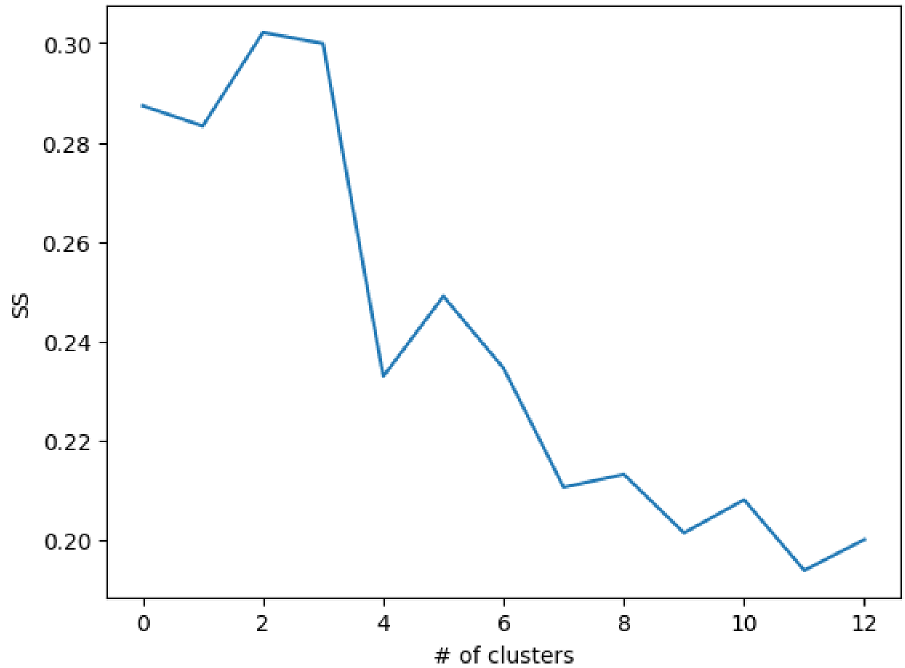


6.) Do the same for a silhoutte plot

```
In [ ]: from sklearn.metrics import silhouette_score
```

```
In [ ]: SSs = []
        Ks = range(2,15)
        for k in Ks:
            kmeans = KMeans(n_clusters = k, n_init=30).fit(X_scaled)
            sil = silhouette_score(X_scaled, kmeans.labels_)
            SSs.append(sil)
```

```
In [ ]: plt.plot(SSs)
        plt.xlabel('# of clusters')
        plt.ylabel('SS')
        plt.show()
```



7.) Create a list of the countries that are in each cluster. Write interesting things you notice.

In [ ]:
```
kmeans = KMeans(n_clusters = 2, n_init=30).fit(X_scaled)
```

In [ ]:
```
preds = pd.DataFrame(kmeans.labels_)
preds
```

Out[ ]:

	0
0	0
1	1
2	1
3	0
4	1
...	...
162	0
163	1
164	1
165	0
166	0

167 rows × 1 columns

In [ ]:
```
output = pd.concat([preds, df],axis = 1)
output
```

Out[ ]:

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

167 rows × 11 columns

In [ ]:
```
print('Cluster1:' )
list(output.loc[output[0] == 0,'country'])
```

```
Cluster1:
```

Out[ ]:

```
['Afghanistan',
 'Angola',
 'Bangladesh',
 'Benin',
 'Bolivia',
 'Botswana',
 'Burkina Faso',
 'Burundi',
 'Cambodia',
 'Cameroon',
 'Central African Republic',
 'Chad',
 'Comoros',
 'Congo, Dem. Rep.',
 'Congo, Rep.',
 "Cote d'Ivoire",
 'Egypt',
 'Equatorial Guinea',
 'Eritrea',
 'Gabon',
 'Gambia',
 'Ghana',
 'Guatemala',
 'Guinea',
 'Guinea-Bissau',
 'Guyana',
 'Haiti',
 'India',
 'Indonesia',
 'Iraq',
 'Kenya',
 'Kiribati',
 'Kyrgyz Republic',
 'Lao',
 'Lesotho',
 'Liberia',
 'Madagascar',
 'Malawi',
 'Mali',
 'Mauritania',
 'Micronesia, Fed. Sts.',
 'Mongolia',
 'Mozambique',
 'Myanmar',
 'Namibia',
 'Nepal',
 'Niger',
 'Nigeria',
 'Pakistan',
 'Philippines',
 'Rwanda',
 'Samoa',
 'Senegal',
 'Sierra Leone',
 'Solomon Islands',
 'South Africa',
 'Sudan',
 'Tajikistan',
 'Tanzania',
 'Timor-Leste',
 'Togo',
 'Tonga',
 'Turkmenistan',
 'Uganda',
 'Uzbekistan',
 'Vanuatu',
 'Yemen',
 'Zambia']
```

In [ ]:

```
print('Cluster2:' )
list(output.loc[output[0] == 1,'country'])
```

```
Cluster2:
```

```
Out[ ]: ['Albania',
 'Algeria',
 'Antigua and Barbuda',
 'Argentina',
 'Armenia',
 'Australia',
 'Austria',
 'Azerbaijan',
 'Bahamas',
 'Bahrain',
 'Barbados',
 'Belarus',
 'Belgium',
 'Belize',
 'Bhutan',
 'Bosnia and Herzegovina',
 'Brazil',
 'Brunei',
 'Bulgaria',
 'Canada',
 'Cape Verde',
 'Chile',
 'China',
 'Colombia',
 'Costa Rica',
 'Croatia',
 'Cyprus',
 'Czech Republic',
 'Denmark',
 'Dominican Republic',
 'Ecuador',
 'El Salvador',
 'Estonia',
 'Fiji',
 'Finland',
 'France',
 'Georgia',
 'Germany',
 'Greece',
 'Grenada',
 'Hungary',
 'Iceland',
 'Iran',
 'Ireland',
 'Israel',
 'Italy',
 'Jamaica',
 'Japan',
 'Jordan',
 'Kazakhstan',
 'Kuwait',
 'Latvia',
 'Lebanon',
 'Libya',
 'Lithuania',
 'Luxembourg',
 'Macedonia, FYR',
 'Malaysia',
 'Maldives',
 'Malta',
 'Mauritius',
 'Moldova',
 'Montenegro',
 'Morocco',
 'Netherlands',
 'New Zealand',
 'Norway',
 'Oman',
 'Panama',
 'Paraguay',
 'Peru',
 'Poland',
 'Portugal',
 'Qatar',
 'Romania',
 'Russia',
 'Saudi Arabia',
 'Serbia',
 'Seychelles',
 'Singapore',
```

```
'Slovak Republic',
'Slovenia',
'South Korea',
'Spain',
'Sri Lanka',
'St. Vincent and the Grenadines',
'Suriname',
'Sweden',
'Switzerland',
'Thailand',
'Tunisia',
'Turkey',
'Ukraine',
'United Arab Emirates',
'United Kingdom',
'United States',
'Uruguay',
'Venezuela',
'Vietnam']
```

In [ ]:
```
#### Write an observation
```

It appears that countries are categorized into developed and developing categories, or they are categorized by GDPs. Cluster 1 comprises nations such as Afghanistan, Bangladesh, Congo, which generally exhibit lower GDPs and economic performances. On the other hand, Cluster 2 includes countries like Australia, Canada, Germany, Japan, which typically have higher GDPs.

8.) Create a table of Descriptive Statistics. Rows being the Cluster number and columns being all the features. Values being the mean of the centroid. Use the nonscaled X values for interprotation

In [ ]:
```
output.drop('country',axis = 1)
```

Out[ ]:

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

167 rows × 10 columns

In [ ]:
```
Q8DF = pd.concat([preds,X], axis = 1)
```

In [ ]:
```
group = Q8DF.groupby(0)
group.mean()
```

Out[ ]:

0	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	76.280882	30.198515	6.090147	43.642146	4227.397059	11.098750	61.910294	4.413824	1981.235294
1	12.161616	48.603030	7.314040	49.121212	26017.171717	5.503545	76.493939	1.941111	20507.979798

In [ ]:
```
group.std()
```

Out[ ]:

0	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	38.076068	18.201742	2.645319	19.323451	4890.581414	13.682630	6.897418	1.285590	2528.509189
1	8.523122	30.116032	2.716652	26.928785	20441.749847	6.957187	3.735757	0.486744	20578.727127

In [ ]:

9.) Write an observation about the descriptive statistics.

We categorized the groups based on the index (0 or 1), which could represent the GDP or economic status of a country, indicating developed and developing countries. From the findings, we observe that child mortality rate, inflation, and total fertility rate are higher in developing countries compared to developed ones, whereas other economic indicators and health expenditure show the opposite trend. Additionally, we note that the standard error of economic factors is higher in developed countries than in developing ones, whereas the standard error of child mortality rate, inflation, life expectancy, and total fertility rate is higher in developing countries than in developed ones.