ml lab homework 9

March 7, 2024

1 Machine Learning Lab Homework 9

2 Connor O'Keefe

3 03/07/2024

4 0.) Import and Clean data

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
```

[2]:
```
df = pd.read_csv("Country-data.csv", sep = ",")
```

[3]:
```
df
```

[3]:
```
                 country  child_mort  exports  health  imports  income  \
0            Afghanistan        90.2     10.0    7.58     44.9    1610
1                Albania        16.6     28.0    6.55     48.6    9930
2                Algeria        27.3     38.4    4.17     31.4   12900
3                 Angola       119.0     62.3    2.85     42.9    5900
4    Antigua and Barbuda        10.3     45.5    6.03     58.9   19100
..                   …           …        …       …        …       …
162              Vanuatu        29.2     46.6    5.25     52.7    2950
163            Venezuela        17.1     28.5    4.91     17.6   16500
164              Vietnam        23.3     72.0    6.84     80.2    4490
165                Yemen        56.3     30.0    5.18     34.4    4480
166               Zambia        83.1     37.0    5.89     30.9    3280

     inflation  life_expec  total_fer   gdpp
0         9.44        56.2       5.82    553
1         4.49        76.3       1.65   4090
2        16.10        76.5       2.89   4460
3        22.40        60.1       6.16   3530
```

```
4         1.44        76.8        2.13  12200
..         ...         ...       ...    ...
162        2.62        63.0        3.50   2970
163       45.90        75.4        2.47  13500
164       12.10        73.1        1.95   1310
165       23.60        67.5        4.67   1310
166       14.00        52.0        5.40   1460

[167 rows x 10 columns]
```

[4]:
```
X = df.drop('country', axis=1)
scaler = StandardScaler().fit(X)
X_scaled = scaler.transform(X)
```

Question we want to answer: Can k-means identify developing economies?

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

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

```
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

6 2.) Pick two features to visualize across

[6]:
```
X.columns
```

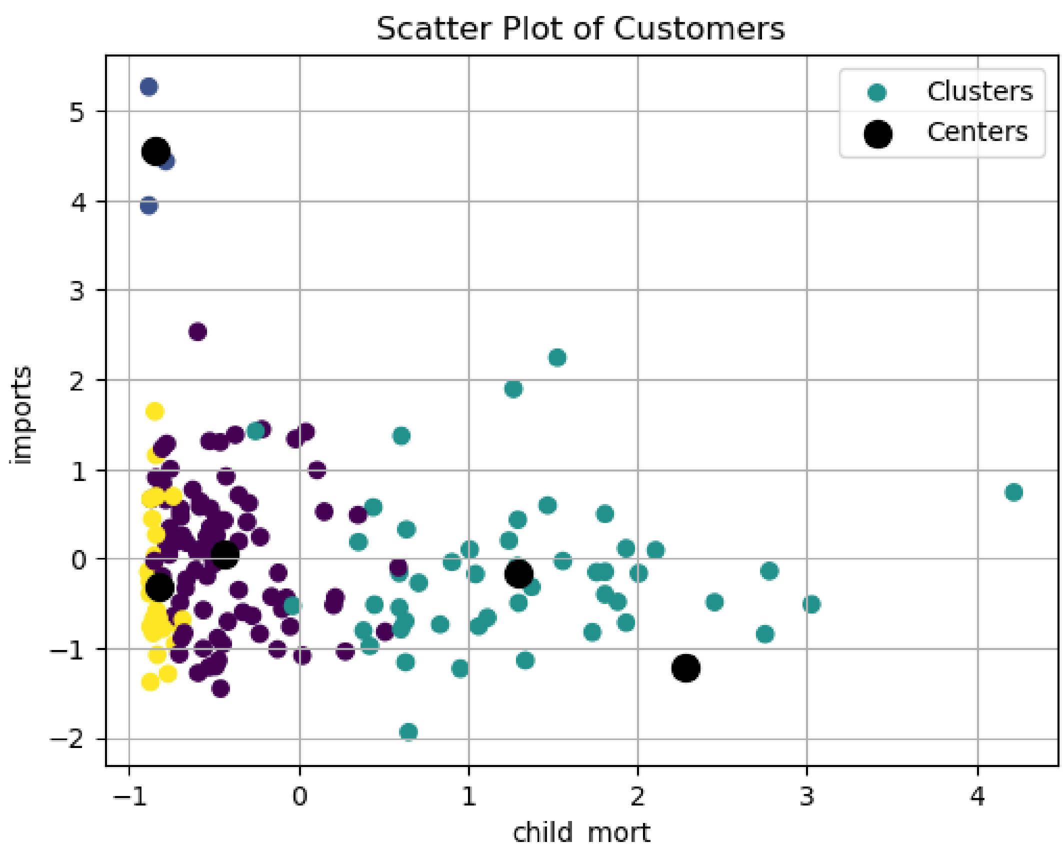
[6]:
```
Index(['child_mort', 'exports', 'health', 'imports', 'income', 'inflation',
       'life_expec', 'total_fer', 'gdpp'],
      dtype='object')
```

[7]:
```
# picking what variables to compare below
# child mortality vs imports
x1_index = 0
x2_index = 3

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

centers = plt.scatter(kmeans.cluster_centers_[:, x1_index], kmeans.
 ↪cluster_centers_[:, x2_index], marker='o', color='black', s=100,␣
 ↪label='Centers')
```

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



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

[8]:
```
WCSSs = []
Ks = range(1,15)
for k in Ks:
    kmeans = KMeans(n_clusters = k, n_init = 30).fit(X_scaled)
    WCSSs.append(kmeans.inertia_)
```

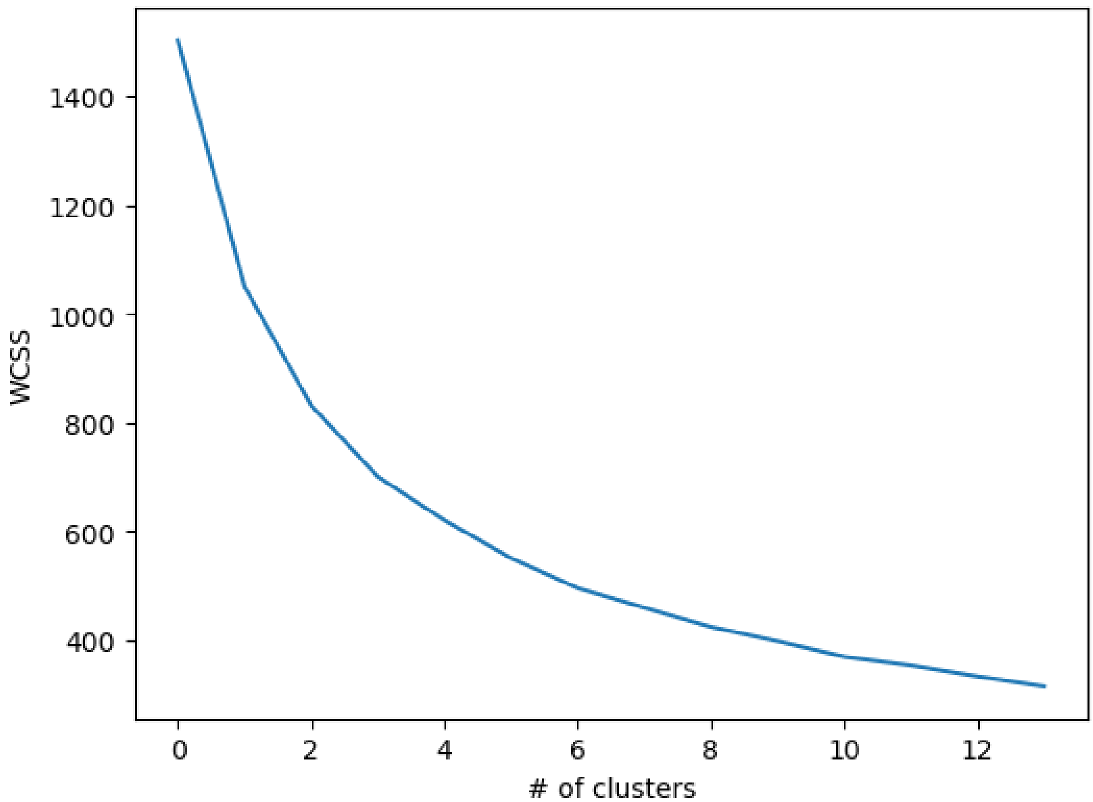
```
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
```

```
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
```

```
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

8 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.

[10]:
```
plt.plot(WCSSs)
plt.xlabel('# of clusters')
plt.ylabel('WCSS')
plt.show()
```



As can be seen above, there is no real elbow. This does not mean that the method is not working, just that the ideal number of clusters is up for interpretation and dependent on the context. Choosing two clusters for comparison seems to be an ideal number, because we could compare affluent countries to impoverished countries.

9 5.) Do the same for a silhoutte plot

[11]:
```
from sklearn.metrics import silhouette_score
```

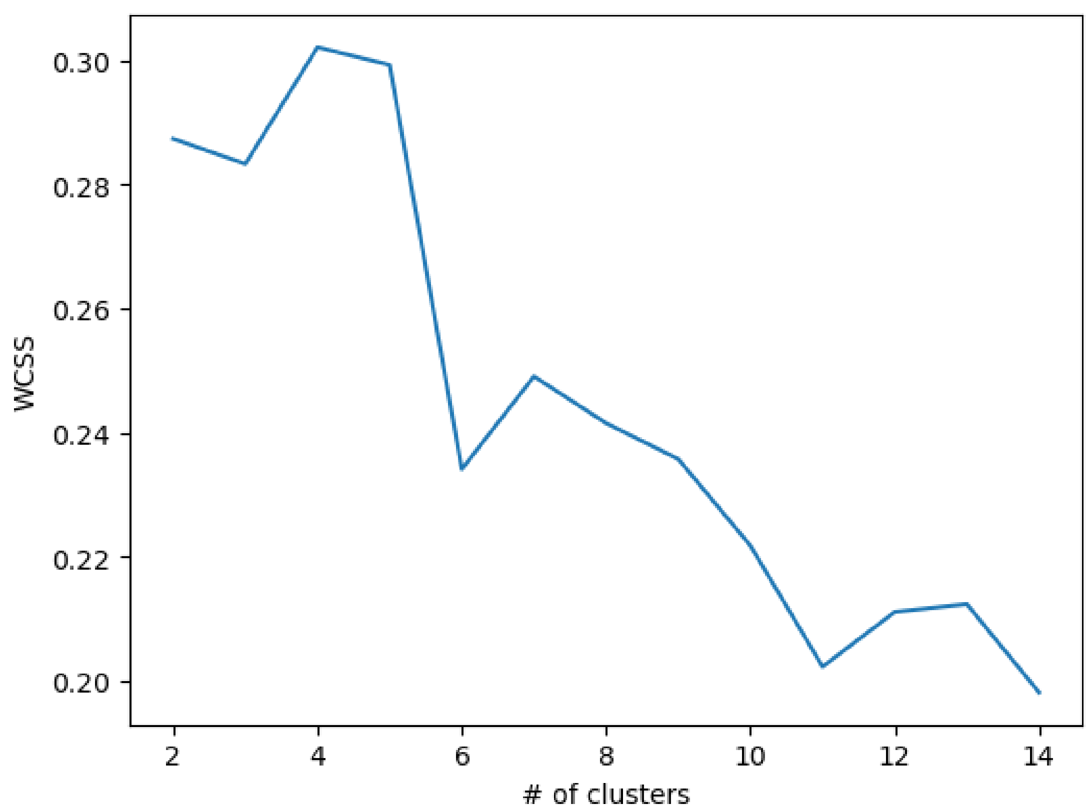
[12]:
```
SSs = []
Ks = range(2,15) # can't only have one cluster because no other cluster to␣
↪compare with
for k in Ks:
    kmeans = KMeans(n_clusters = k, n_init = 30).fit(X_scaled)
    sil = silhouette_score(X_scaled, kmeans.labels_)
    SSs.append(sil)
```

```
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
```

```
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
```

```
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

[13]:
```
plt.plot(Ks, SSs)
plt.xlabel('# of clusters')
plt.ylabel('WCSS')
plt.show()
```



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

[14]:
```
# choosing two clusters below
kmeans = KMeans(n_clusters = 2, n_init = 30).fit(X_scaled)
```

```
C:\Users\12282\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1446:
UserWarning: KMeans is known to have a memory leak on Windows with MKL, when
there are less chunks than available threads. You can avoid it by setting the
environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

[15]:
```
preds = pd.DataFrame(kmeans.labels_)
```

[16]:
```
output = pd.concat([preds, df], axis = 1)
output
```

[16]:
```
     0            country  child_mort  exports  health  imports  income  \
0    0        Afghanistan        90.2     10.0    7.58     44.9    1610
1    1            Albania        16.6     28.0    6.55     48.6    9930
2    1            Algeria        27.3     38.4    4.17     31.4   12900
3    0             Angola       119.0     62.3    2.85     42.9    5900
4    1  Antigua and Barbuda        10.3     45.5    6.03     58.9   19100
..  ..                 …           …       …       …        …       …
162  0            Vanuatu        29.2     46.6    5.25     52.7    2950
163  1          Venezuela        17.1     28.5    4.91     17.6   16500
164  1            Vietnam        23.3     72.0    6.84     80.2    4490
165  0              Yemen        56.3     30.0    5.18     34.4    4480
166  0             Zambia        83.1     37.0    5.89     30.9    3280

     inflation  life_expec  total_fer   gdpp
0         9.44        56.2       5.82    553
1         4.49        76.3       1.65   4090
2        16.10        76.5       2.89   4460
3        22.40        60.1       6.16   3530
4         1.44        76.8       2.13  12200
..         …           …        …       …
162       2.62        63.0       3.50   2970
163      45.90        75.4       2.47  13500
164      12.10        73.1       1.95   1310
165      23.60        67.5       4.67   1310
166      14.00        52.0       5.40   1460

[167 rows x 11 columns]
```

[17]:
```
cluster1 = list(output.loc[output[0] == 0, 'country'])
print('Cluster 1: ', cluster1)
```

```
Cluster 1:  ['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']
```

[18]:
```
cluster2 = list(output.loc[output[0] == 1, 'country'])
print('Cluster 2: ', cluster2)
```

```
Cluster 2:  ['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']
```

Because I am an American, my first point of interest was indentifying how the United States was classified (Cluster 2). Overall, Cluster 2 seems to classify the more developed countries and Cluster 1 seems to classify developing countries, with a fair amount from Africa and Asia. More analysis follows below.

11 7.) 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

[21]:
```
output.drop('country', axis = 1).groupby(0).mean()
```

```
[21]:    child_mort    exports    health    imports        income  inflation  \
0
0   76.280882  30.198515  6.090147  43.642146   4227.397059  11.098750
1   12.161616  48.603030  7.314040  49.121212  26017.171717   5.503545

   life_expec  total_fer          gdpp
0
0   61.910294   4.413824   1981.235294
1   76.493939   1.941111  20507.979798
```

```
[20]: output.drop('country', axis = 1).groupby(0).std()
```

```
[20]:    child_mort    exports    health    imports        income  inflation  \
0
0   38.076068  18.201742  2.645319  19.323451   4890.581414  13.682630
1    8.523122  30.116032  2.716652  26.928785  20441.749847   6.957187

   life_expec  total_fer          gdpp
0
0    6.897418   1.285590   2528.509189
1    3.735757   0.486744  20578.727127
```

12 8.) Write an observation about the descriptive statistics.

As I hypothesized above, I think Cluster 1 may classify developing countries and Cluster 2 may classify developed countries. So now, turning to the above statistics, some important differences can be noticed. Every “good” and “bad” mean statistic is more optimal for Cluster 2, which supports my claim. For example, mean income of Cluster 2 is more than 6x that of Cluster 1. The standard deviation statistics are a little bit harder to interpret, and can be for a few different reasons. For example, the standard deviation of inflation of the developing countries could be higher because there is a huge variation in inflation rates, impacted by countries like Sri Lanka and Venezuela.