

▼ 0.) Import and Clean data

```
import pandas as pd
from google.colab import drive
import matplotlib.pyplot as plt
import numpy as np
```

```
from sklearn.preprocessing import StandardScaler
```

```
import seaborn as sns
from sklearn.decomposition import PCA
```

```
drive.mount('/content/gdrive/', force_remount = True)
```

```
Mounted at /content/gdrive/
```

```
df = pd.read_csv("/content/gdrive/MyDrive/Econ441B/Country-data.csv", sep = ",")
```

```
df.head()
```

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer
0	Afghanistan	90.2	10.0	7.58	44.9	1610	9.44	56.2	5.82
1	Albania	16.6	28.0	6.55	48.6	9930	4.49	76.3	1.65
2	Algeria	27.3	38.4	4.17	31.4	12900	16.10	76.5	2.89
3	Angola	119.0	62.3	2.85	42.9	5900	22.40	60.1	6.16
4	Antigua and and	10.3	45.5	6.03	58.9	19100	1.44	76.8	2.13

```
df.columns
```

```
Index(['country', 'child_mort', 'exports', 'health', 'imports', 'income',  
      'inflation', 'life_expec', 'total_fer', 'gdp'],  
      dtype='object')
```

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

```
scaler = StandardScaler().fit(X)  
X_scaled = scaler.transform(X)
```

▼ 1.) Run a PCA Algorithm to get 2 Principle Components for the 9 X features

```
pca = PCA(n_components = 2)  
X_pca = pca.fit_transform(X_scaled)
```

```
X_pca  
# 2 columns of new x points converted from the 9 above
```

```
[ 1.814006/48e+00, -1.45088054e+00],
[ 4.24229634e+00, -1.95603674e-01],
[ 5.72792704e-01, -6.37384843e-01],
[ 1.63761544e-01, -1.06667848e+00],
[-1.67970356e+00, -1.00162862e+00],
[-5.62897632e-01, -2.21043960e-02],
[ 8.55935813e-01, -1.83440759e-01],
[-1.91217031e+00, 9.15599347e-02],
[ 8.32420187e-01, -8.69325996e-01],
[ 1.60259775e+00, 2.93912057e+00],
[-3.38162479e+00, -2.36301516e-01],
[ 5.78337630e+00, 6.68209028e+00],
[ 2.02972370e+00, 1.05040745e+00],
[ 2.27949171e+00, 1.95275226e-01],
[-8.06209136e-01, 1.30349059e+00],
[-1.19183736e+00, -5.56757164e-01],
[ 1.91806245e+00, -4.27468245e-01],
[ 2.01919721e+00, -1.78438246e+00],
[-5.75572155e-01, -9.97551478e-01],
[ 2.66234652e-02, -1.60640815e-02],
[-2.31942387e+00, -7.69407328e-01],
[ 1.71674731e-01, -9.48076409e-02],
[ 2.81832286e+00, -9.14480968e-01],
[ 4.08854413e+00, -4.29461909e-01],
[-1.24446436e+00, -2.89174316e-02],
[-2.55404919e+00, -2.15027956e-01],
[ 9.26092707e-01, 8.28230655e-01],
[-2.37197047e+00, -1.17751295e+00],
[-1.99764225e+00, 9.58361586e-01],
[-7.55008538e-01, -8.78938568e-02],
[ 6.02231612e-01, 1.73435708e-01],
[ 4.01437705e-01, -1.41198973e+00],
[-4.63936165e-01, 1.29187347e+00],
[-2.85483624e+00, -3.52082382e-01],
[ 3.02299800e-01, -9.75710669e-02],
[ 2.42714125e+00, 1.15181307e+00],
[ 2.06798993e+00, -1.53531349e+00],
[ 2.64120583e+00, -2.99736446e+00],
[ 6.17312598e-01, -1.43047723e+00],
[-8.53528944e-01, -6.54485112e-01],
[-8.20631131e-01, 6.39570072e-01],
[-5.51035564e-01, -1.23388618e+00],
[ 4.98524385e-01, 1.39074432e+00],
[-1.88745106e+00, -1.09453015e-01],
[-2.86406392e+00, 4.85997985e-01]]
```

```
X_pca[:,0]
```

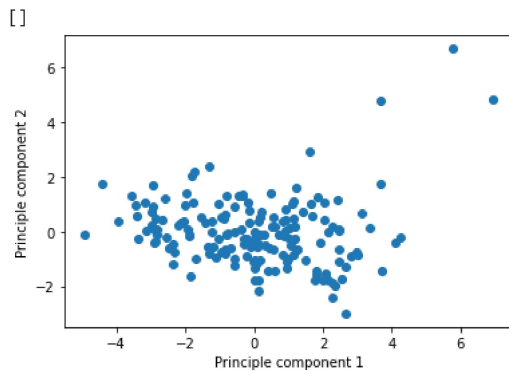
```
#every row and 1rst column
```

```
array([-2.91302459e+00, 4.29911330e-01, -2.85225077e-01, -2.93242265e+00,
        1.03357587e+00, 2.24072616e-02, -1.01583737e-01, 2.34216461e+00,
        2.97376366e+00, -1.81486997e-01, 1.26874386e+00, 1.67099640e+00,
        -1.12385093e+00, 1.08137420e+00, 5.80025152e-01, 3.14378596e+00,
        2.11255447e-01, -2.67231388e+00, -1.56570962e-01, -7.93851561e-01,
        9.95867143e-01, -8.82087639e-01, 1.40781361e-01, 2.46008609e+00,
        9.06594515e-01, -3.12205344e+00, -2.89897068e+00, -5.82411867e-01,
        -2.80790857e+00, 2.54363055e+00, -1.55801452e-01, -3.96496402e+00,
        -3.55755520e+00, 9.51656055e-01, 5.74819803e-02, 1.21146120e-01,
        -2.09355643e+00, -3.17337012e+00, -1.72567641e+00, 9.37826615e-01,
        -2.58170623e+00, 1.14886344e+00, 2.17445492e+00, 2.05326329e+00,
        3.01049182e+00, -2.31102923e-01, 9.61833240e-03, -8.48186699e-01,
        8.18678445e-02, -1.29342284e+00, -2.47469590e+00, 1.65908340e+00,
        -1.88828409e-01, 2.45896019e+00, 2.25427080e+00, -1.42171455e+00,
        -2.2136958e+00, 3.21942207e-01, 2.67142195e+00, -2.05416693e+00,
        1.77949294e+00, 1.45504799e-01, -6.63503125e-01, -2.96952947e+00,
        -2.83361647e+00, -3.22781465e-01, -4.40971727e+00, 1.83916013e+00,
        2.48092396e+00, -1.34282579e+00, -9.54750124e-01, -1.06461193e-03,
        -1.02922816e+00, 3.66862804e+00, 1.48531666e+00, 2.16580995e+00,
        1.86093002e-02, 2.26588199e+00, 1.60142643e-01, -2.93346500e-01,
        -1.87470247e+00, -1.23921686e+00, 2.46565870e+00, -3.39969880e-01,
        -1.52776995e+00, 1.18883984e+00, 1.17199076e+00, -1.80315140e+00,
        -1.77358023e+00, 8.18943051e-01, 1.40978812e+00, 6.91775496e+00,
        7.33210319e-01, -2.13600867e+00, -2.97988525e+00, 1.23082842e+00,
        1.10860101e+00, -3.41225513e+00, 3.67954260e+00, -1.95392747e+00,
        8.99775055e-01, -3.80928795e-01, 5.09539453e-01, -9.44975538e-01,
        1.02668389e+00, -2.32870156e-01, -2.92054051e+00, -1.83719774e+00,
        -1.04337471e+00, -1.30708985e+00, 3.37915727e+00, 1.81574666e+00,
        -3.45016774e+00, -4.91206615e+00, 3.72119513e+00, 1.12738665e+00,
        -2.36034718e+00, 1.16378429e+00, 1.17846224e-01, -2.06354519e-02,
        -7.82745871e-01, 1.21782754e+00, 1.81406748e+00, 4.24229634e+00,
        5.72792704e-01, 1.63761544e-01, -1.67970356e+00, -5.62897632e-01,
        8.55935813e-01, -1.91217031e+00, 8.32420187e-01, 1.60259775e+00,
        -3.38162479e+00, 5.78337630e+00, 2.02972370e+00, 2.27949171e+00,
        -8.06209136e-01, -1.19183736e+00, 1.91806245e+00, 2.01919721e+00,
```

```
-5.75572155e-01, 2.66234652e-02, -2.31942387e+00, 1.71674731e-01,
2.81832286e+00, 4.08854413e+00, -1.24446436e+00, -2.55404919e+00,
9.26092707e-01, -2.37197047e+00, -1.99764225e+00, -7.55008538e-01,
6.02231612e-01, 4.01437705e-01, -4.63936165e-01, -2.85483624e+00,
3.02299800e-01, 2.42714125e+00, 2.06798993e+00, 2.64120583e+00,
6.17312598e-01, -8.53528944e-01, -8.20631131e-01, -5.51035564e-01,
4.98524385e-01, -1.88745106e+00, -2.86406392e+00]]
```

2.) Plot a Scatter plot of the PCs on the axis

```
plt.scatter(x=X_pca[:,0], y = X_pca[:,1])
plt.xlabel('Principle component 1')
plt.ylabel('Principle component 2')
plt.plot()
```



3.) Rank the features in order of importance according to PCA

```
loadings = pca.components_
```

```
feature_importance = pd.DataFrame(np.sum(loadings**2, axis = 0))
# 1st list is pca 1, 2nd is pca 2
# add them together based on the rows
```

```
feature_names = df.columns[1:]
```

```
feature_importance.index = feature_names
```

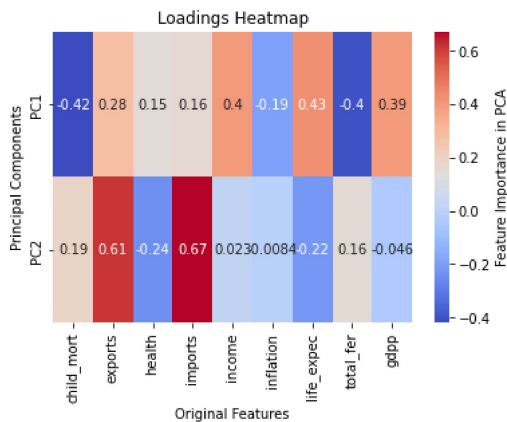
```
feature_importance.sort_values(0, ascending = False)
# want most imp on top
```

	0 
imports	0.477420
exports	0.456567
life_expec	0.230937
child_mort	0.213201
total_fer	0.187094
income	0.159263
gdpp	0.156288
health	0.081843
inflation	0.037386

4.) Plot a heatmap of the feature importance (Fill in all parameters)

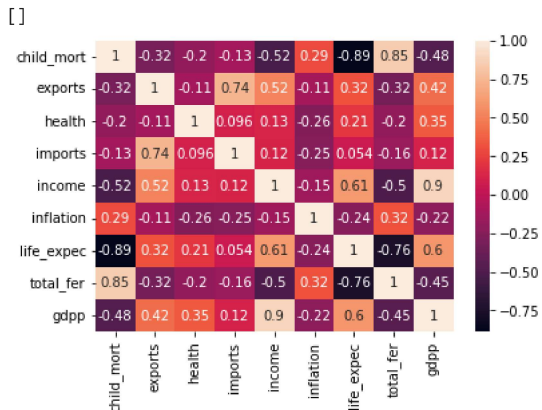
```
feature_names = df.columns[1:]
```

```
#sns.heatmap(, annot=, cmap='coolwarm', xticklabels=, yticklabels=[, ], cbar_kws={'label':''})
sns.heatmap(loadings, annot = True, cmap='coolwarm', xticklabels= feature_names, yticklabels=["PC1","PC2" ], cbar_kws={'label':'Feature Import'})
plt.xlabel('Original Features')
plt.ylabel('Principal Components')
plt.title('Loadings Heatmap')
plt.show()
```



5.) Plot a correlation plot of the original features. What do you notice between the graphs of 4 & 5?

```
sns.heatmap(X.corr(), annot = True)
plt.plot()
```



- After analyzing the graphs for 4 and 5, I noticed an important finding: if two variables have a high correlation, it appears that they will have greater feature importance. For example, this can be seen when analyzing the visualizations for exports and imports for both graphs.

6.) Run a PCA with 9 PCs. Plot a Cumulative Explained Variance Plot. How many PCs should we use if we want to retain 95% of the variance?

```
pca = PCA(n_components = 9)
X_pca = pca.fit_transform(X_scaled)
```

```
np.cumsum(pca.explained_variance_ratio_)
# amount the principle component explains variance for each feature
```

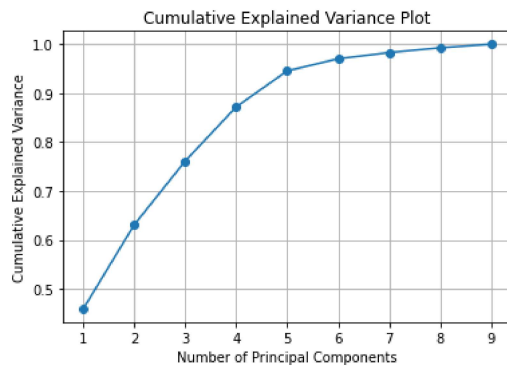
```
array([0.4595174 , 0.17181626, 0.13004259, 0.11053162, 0.07340211,
       0.02484235, 0.0126043 , 0.00981282, 0.00743056])
```

```

cumulative_explained_variance = np.cumsum(pca.explained_variance_ratio_)

plt.plot(np.arange(1, len(cumulative_explained_variance) + 1), cumulative_explained_variance, marker='o')
plt.xlabel('Number of Principal Components')
plt.ylabel('Cumulative Explained Variance')
plt.title('Cumulative Explained Variance Plot')
plt.grid()
plt.show()

```



```

cumulative_explained_variance
array([0.4595174 , 0.63133365, 0.76137624, 0.87190786, 0.94530998,
       0.97015232, 0.98275663, 0.99256944, 1.          ])

```

- We need at least 6 principle components to retain 95% of the variance