```
import pandas as pd
import numpy as np

from matplotlib import pyplot as plt

df = pd.read_csv("C:/Users/David/Downloads/paises_mundo.csv")

df.head()
```

```
   CrecPobl  MortInf  PorcMujeres   PNB95  ProdElec  LinTelf  ConsAgua
\
0       1.0       30           41    2199      3903       12        94

1       3.0      124           46    4422       955        6        57

2       4.3       21           13  133540     91019       96       497

3       2.5       34           24   44609     19883       42       180

4       1.3       22           31  278431     65962      160      1043


   PropBosq  PropDefor  ConsEner  EmisCO2
0        53        0.0       341      1.2
1        19        0.7        89      0.5
2         1        0.0      4566     13.1
3         2        0.8       906      3.0
4        22        0.1      1504      3.5
```

Parte 1

1. Calcule las matrices de varianza-covarianza S con cov(X) y la matriz de correlaciones con cor(X)

```
df_numerico = df.select_dtypes(include=['float64', 'int64'])

cov = df_numerico.cov()

cor = df_numerico.corr()

cov, cor
```

```
(                CrecPobl       MortInf    PorcMujeres         PNB95
\
 CrecPobl        1.538298  2.195026e+01      -6.078026 -8.933379e+04

 MortInf        21.950263  1.032859e+03      -9.249342 -2.269332e+06

 PorcMujeres    -6.078026 -9.249342e+00      76.983224  2.813114e+05
```

```
PNB95       -89333.788772 -2.269332e+06  281311.418421  4.999786e+10
ProdElec    -49739.635746 -1.043435e+06  226024.813487  2.247791e+10
LinTelf       -136.907895 -4.381366e+03     449.975000  2.039550e+07
ConsAgua       -48.270921 -1.288211e+03   -1568.313487  1.097481e+07
PropBosq        -3.887018 -1.466316e+01      65.178947  2.474311e+05
PropDefor        0.336197  1.276296e+01       0.268059 -5.806203e+04
ConsEner      -838.416886 -4.442568e+04     285.520724  1.415628e+08
EmisCO2         -1.137877 -9.485500e+01      -2.150132  2.501673e+05

                  ProdElec       LinTelf      ConsAgua       PropBosq
\
CrecPobl    -4.973964e+04 -1.369079e+02 -4.827092e+01      -3.887018
MortInf     -1.043435e+06 -4.381366e+03 -1.288211e+03     -14.663158
PorcMujeres  2.260248e+05  4.499750e+02 -1.568313e+03      65.178947
PNB95        2.247791e+10  2.039550e+07  1.097481e+07  247431.122807
ProdElec     1.821909e+10  7.583050e+06  1.399817e+07   70359.785965
LinTelf      7.583050e+06  3.841247e+04  1.193110e+04     248.715789
ConsAgua     1.399817e+07  1.193110e+04  3.301981e+05   -2220.757895
PropBosq     7.035979e+04  2.487158e+02 -2.220758e+03     401.003509
PropDefor   -3.180340e+04 -9.940461e+01 -6.743793e+01       2.625263
ConsEner     6.801296e+07  3.426262e+05  2.092242e+05   -5153.438596
EmisCO2      1.392779e+05  6.385700e+02  4.869328e+02     -12.897193

                PropDefor      ConsEner        EmisCO2
CrecPobl         0.336197 -8.384169e+02      -1.137877
MortInf         12.762961 -4.442568e+04     -94.855000
PorcMujeres      0.268059  2.855207e+02      -2.150132
PNB95       -58062.027632  1.415628e+08  250167.323509
ProdElec    -31803.401546  6.801296e+07  139277.888640
LinTelf        -99.404605  3.426262e+05     638.570000
ConsAgua       -67.437928  2.092242e+05     486.932763
PropBosq         2.625263 -5.153439e+03     -12.897193
```

```
 PropDefor        1.817253 -1.051522e+03      -2.632487
 ConsEner     -1051.521875  5.014395e+06   10286.159781
 EmisCO2         -2.632487  1.028616e+04      27.268614  ,
              CrecPobl   MortInf  PorcMujeres     PNB95  ProdElec
LinTelf  \
 CrecPobl     1.000000  0.550679    -0.558527 -0.322122 -0.297111 -
0.563212
 MortInf      0.550679  1.000000    -0.032801 -0.315792 -0.240537 -
0.695589
 PorcMujeres -0.558527 -0.032801     1.000000  0.143388  0.190851
0.261670
 PNB95       -0.322122 -0.315792     0.143388  1.000000  0.744761
0.465396
 ProdElec    -0.297111 -0.240537     0.190851  0.744761  1.000000
0.286645
 LinTelf     -0.563212 -0.695589     0.261670  0.465396  0.286645
1.000000
 ConsAgua    -0.067730 -0.069756    -0.311062  0.085415  0.180477
0.105939
 PropBosq    -0.156503 -0.022784     0.370967  0.055259  0.026031
0.063371
 PropDefor    0.201079  0.294593     0.022663 -0.192623 -0.174784 -
0.376238
 ConsEner    -0.301877 -0.617311     0.014532  0.282725  0.225019
0.780684
 EmisCO2     -0.175689 -0.565208    -0.046928  0.214251  0.197600
0.623937

              ConsAgua  PropBosq  PropDefor  ConsEner   EmisCO2
 CrecPobl    -0.067730 -0.156503   0.201079 -0.301877 -0.175689
 MortInf     -0.069756 -0.022784   0.294593 -0.617311 -0.565208
 PorcMujeres -0.311062  0.370967   0.022663  0.014532 -0.046928
 PNB95        0.085415  0.055259  -0.192623  0.282725  0.214251
 ProdElec     0.180477  0.026031  -0.174784  0.225019  0.197600
 LinTelf      0.105939  0.063371  -0.376238  0.780684  0.623937
 ConsAgua     1.000000 -0.192992  -0.087058  0.162598  0.162274
 PropBosq    -0.192992  1.000000   0.097250 -0.114925 -0.123336
 PropDefor   -0.087058  0.097250   1.000000 -0.348338 -0.373962
 ConsEner     0.162598 -0.114925  -0.348338  1.000000  0.879655
 EmisCO2      0.162274 -0.123336  -0.373962  0.879655  1.000000  )
```

1. Calcule los valores y vectores propios de cada matriz.La función en R es: eigen().

```
eigen_cov_valores, eigen_cov_vectores = np.linalg.eig(cov)

eigen_cor_valores, eigen_cor_vectores = np.linalg.eig(cor)

eigen_cov_valores, eigen_cov_vectores, eigen_cor_valores,
eigen_cor_vectores
```

```
(array([6.16357629e+10, 6.58161227e+09, 4.63625593e+06,
3.10723182e+05,
        1.21601494e+04, 5.13776704e+02, 3.62788506e+02,
4.54208135e+01,
        5.80086834e+00, 4.76808277e-01, 1.43801991e+00]),
 array([[-1.65816773e-06,  4.70678460e-07, -1.26373574e-04,
         -1.92840781e-05,  5.53739707e-03,  1.24345612e-02,
         -5.35908938e-03, -8.39081045e-02, -6.77835776e-02,
         -9.87288738e-01, -1.15809094e-01],
        [-4.04813855e-05, -1.77425442e-05, -8.22538212e-03,
         -2.49325727e-03,  9.44030203e-02,  9.91751509e-01,
         -2.25801959e-02, -7.89112785e-02, -1.63783592e-02,
          2.09249092e-02,  4.26487142e-04],
        [ 5.73909610e-06, -1.08454281e-05, -1.31814909e-04,
          5.53830717e-03, -3.14036410e-02,  8.55299154e-02,
          1.13648080e-01,  9.85649803e-01, -1.46846382e-02,
         -8.34432428e-02,  8.24146556e-03],
        [ 8.88037597e-01,  4.59763191e-01, -2.60220711e-03,
         -3.89358789e-04,  3.32740919e-04, -8.62100480e-06,
          7.56647696e-06,  1.21724819e-05, -3.97146871e-07,
         -2.72399680e-07,  4.27445059e-07],
        [ 4.59763574e-01, -8.88040472e-01, -5.69489558e-04,
          1.09630500e-03, -2.20781916e-04,  1.95540800e-05,
         -1.54465759e-05, -2.55899806e-05,  1.05947081e-06,
          2.08685783e-07, -1.35388052e-06],
        [ 3.50434128e-04,  4.01617912e-04,  6.19424889e-02,
          7.64117441e-03, -9.92140449e-01,  9.10962152e-02,
         -4.74868183e-02, -3.41681151e-02, -5.37954917e-03,
         -4.94439599e-04, -3.40942294e-03],
        [ 2.62550815e-04, -1.12211782e-03,  4.01453227e-02,
         -9.99141142e-01, -5.77951444e-03, -1.08722945e-03,
          6.86329425e-03,  4.69873084e-03,  7.96526063e-05,
         -4.78041639e-04,  3.62142483e-05],
        [ 4.08956383e-06,  7.79084284e-06, -1.27199178e-03,
          6.43579663e-03, -4.19331615e-02,  1.72194802e-02,
          9.92053803e-01, -1.16963838e-01,  1.41656634e-03,
          3.74897683e-03,  5.89175813e-03],
        [-1.07382528e-06,  2.35080815e-07, -1.91617666e-04,
          4.04379633e-05,  1.80907507e-03,  1.75866735e-03,
          7.45542650e-03,  1.81144337e-02,  1.28303909e-01,
          1.05293376e-01, -9.85931687e-01],
        [ 2.54715600e-03,  7.12678218e-04,  9.97231550e-01,
          3.97356775e-02,  6.25729475e-02,  2.63967318e-03,
          3.76470692e-03,  1.26705213e-03,  2.26293051e-03,
         -5.90624139e-05,  2.67261848e-04],
        [ 4.64372357e-06, -1.31573121e-06,  2.06790465e-03,
         -5.62604859e-05,  4.23671197e-03, -1.87799378e-02,
          1.70913698e-03, -5.20482299e-03, -9.89152936e-01,
          8.22137089e-02, -1.20051859e-01]]),
 array([4.02987902, 1.92999195, 1.37041115, 0.06935866, 0.14632819,
```

```
       0.16806846, 0.32680096, 0.57130511, 0.86451597, 0.79414057,
       0.72919997]),
 array([[ 0.31411941,  0.34835747,  0.07352541,  0.10062784, -
0.34674573,
         -0.5218122 , -0.09481963,  0.16289743,  0.44028717,
0.32972147,
         -0.18392437],
        [ 0.39239544, -0.04136238,  0.17759254, -0.17487096,
0.3895924 ,
          0.29031618, -0.32307802,  0.63980408,  0.13398483, -
0.08340489,
         -0.0865639 ],
        [-0.11654632, -0.58283641, -0.16686305,  0.167868  , -
0.42854658,
         -0.23599758,  0.05209889,  0.53108671, -0.05865031, -0.186541
,
          0.1683565 ],
        [-0.29539377, -0.17690839,  0.53343025,  0.15247432, -
0.34911534,
          0.36995675, -0.44913216, -0.1490207 ,  0.26248209,
0.14110658,
          0.04653378],
        [-0.25896472, -0.17356372,  0.61438847, -0.12366382,
0.33770404,
         -0.30681318,  0.50343911,  0.10827458,  0.17389644,
0.07521971,
          0.02821905],
        [-0.44608293, -0.02719077, -0.1517725 , -0.44992596,
0.20997673,
         -0.4473311 , -0.56975094, -0.00085016, -0.04959796,
0.05416498,
          0.02442175],
        [-0.0924105 ,  0.32060987,  0.37024258,  0.0706778 , -
0.20561803,
         -0.08358225, -0.0596247 ,  0.23576667, -0.73603097, -
0.02671021,
         -0.3094089 ],
        [-0.00569293, -0.45742697, -0.16480339,  0.0149371 ,
0.08671232,
          0.0743852 ,  0.04275404, -0.08060367, -0.04024882,
0.41531702,
         -0.75356463],
        [ 0.24365229, -0.15408201,  0.02961449, -0.07259619,
0.03209758,
          0.01868615, -0.01607505,  0.01123336, -0.33650345,
0.73261463,
          0.50894232],
        [-0.41502955,  0.23286257, -0.20608749,  0.67912543,
0.36147417,
          0.04339752, -0.05023582,  0.2711228 ,  0.06730166,
```

```
0.23100421,
         0.05806466],
       [-0.37453103,  0.29168698, -0.20631751, -0.46737561, -
0.28779437,
         0.37666244,  0.30978009,  0.33528221,  0.14843513,
0.24028756,
        -0.02809233]]))
```

1. Calcule la proporción de varianza explicada por cada componente. Se sugiere dividir cada lambda entre la varianza total (las lambdas están en eigen(S)[1]). La varianza total es la suma de las varianzas de la diagonal de S. Una forma es sum(diag(S)). La varianza total de los componentes es la suma de los valores propios (es decir, la suma de la varianza de cada componente), sin embargo, si sumas la diagonal de S (es decir, la varianza de cada x), te da el mismo valor (¡comprúebalo!). Recuerda que las combinaciones lineales buscan reproducir la varianza de X.

```
total_varianza_cov = cov.values.diagonal().sum()

varianza_explicada = eigen_cov_valores / total_varianza_cov

sum_eigen_cov = eigen_cov_valores.sum()

varianza_explicada, sum_eigen_cov, total_varianza_cov
```

```
(array([9.03454311e-01, 9.64729842e-02, 6.79580362e-05, 4.55456679e-
06,
        1.78242937e-07, 7.53091641e-09, 5.31773802e-09, 6.65776294e-
10,
        8.50288738e-11, 6.98903483e-12, 2.10784328e-11]),
 68222335252.701385,
 68222335252.70136)
```

```
varianza_explicada[0:2].sum()
```

```
0.9999272955265136
```

La proporcion de la varianza explicada por cada componente se muestra en el primer arreglo, que nos dice que los 2 primeros componentes explican el 99.99% de la varianza.

tembien estamos mostrando que la suma de los valores propios es igual a la suma de la diagonal de S.

1. Acumule los resultados anteriores. (cumsum() puede servirle).

```
varianza_explicada_comulativa = np.cumsum(varianza_explicada)

varianza_explicada_comulativa
```

```
array([0.90345431, 0.9999273 , 0.99999525, 0.99999981, 0.99999999,
       0.99999999, 1.        , 1.        , 1.        , 1.        ,
       1.        ])
```

Como ya lo vimos en el punto anterior aqui tenemos la varianza acumulada donde podemos ver que en solo 2 componentes se explica practicamente toda la varianza.

1. Según los resultados anteriores, ¿qué componentes son los más importantes? ¿qué variables son las que más contribuyen a la primera y segunda componentes principales? ¿por qué lo dice? ¿influyen las unidades de las variables?

```
componente_1_cov_vars = eigen_cov_vectores[:, 0]
componente_2_cov_vars = eigen_cov_vectores[:, 1]

componente_1_cov_indices = np.argsort(np.abs(componente_1_cov_vars))
[::-1]
componente_2_cov_indices = np.argsort(np.abs(componente_2_cov_vars))
[::-1]

componente_1_vars_importantes =
df_numerico.columns[componente_1_cov_indices]
componente_2_vars_importantes =
df_numerico.columns[componente_2_cov_indices]
```

ya que vimos que los primeros 2 componentes explican casi toda la varianza vamos a analisar que variables son importantes para estos componentes:

```
componente_1_vars_importantes
```

```
Index(['PNB95', 'ProdElec', 'ConsEner', 'LinTelf', 'ConsAgua',
'MortInf',
       'PorcMujeres', 'EmisCO2', 'PropBosq', 'CrecPobl', 'PropDefor'],
      dtype='object')
```

Componente 1: - PNB95 - ProdElec - ConsEner - LinTelf - ConsAgua

```
componente_2_vars_importantes
```

```
Index(['ProdElec', 'PNB95', 'ConsAgua', 'ConsEner', 'LinTelf',
'MortInf',
       'PorcMujeres', 'PropBosq', 'EmisCO2', 'CrecPobl', 'PropDefor'],
      dtype='object')
```

Componente 2: - ProdElec - PNB95 - ConsAgua - LinTelf - MortInf

1. Hacer los mismos pasos anteriores, pero con la matriz de correlaciones (se obtiene con cor(x) si x está compuesto por variables numéricas)

```
varianza_explicada_ratio_cor = eigen_cor_valores /
sum(eigen_cor_valores)

varianza_explicada_comulativa_cor =
np.cumsum(varianza_explicada_ratio_cor)

varianza_explicada_ratio_cor, varianza_explicada_comulativa_cor
```

```
(array([0.36635264, 0.17545381, 0.12458283, 0.00630533, 0.01330256,
        0.01527895, 0.02970918, 0.05193683, 0.07859236, 0.0721946 ,
        0.06629091]),
 array([0.36635264, 0.54180645, 0.66638928, 0.67269462, 0.68599718,
        0.70127613, 0.73098531, 0.78292214, 0.8615145 , 0.93370909,
        1.        ]))
```

Aqui tenemos la varianza explicada por componente utilizando la matriz de correlacion, vemos que es muy distinto a la matriz de covarianza

```
componente_1_cor_vars_contribuyentes = eigen_cor_vectores[:, 0]
componente_2_cor_vars_contribuyentes = eigen_cor_vectores[:, 1]

componente_1_cor_indices =
np.argsort(np.abs(componente_1_cor_vars_contribuyentes))[::-1]
componente_2_cor_indices =
np.argsort(np.abs(componente_2_cor_vars_contribuyentes))[::-1]

componente_1_cor_vars_importantes =
df_numerico.columns[componente_1_cor_indices]
componente_2_cor_vars_importantes =
df_numerico.columns[componente_2_cor_indices]

componente_1_cor_vars_importantes, componente_2_cor_vars_importantes
```

```
(Index(['LinTelf', 'ConsEner', 'MortInf', 'EmisCO2', 'CrecPobl',
'PNB95',
        'ProdElec', 'PropDefor', 'PorcMujeres', 'ConsAgua',
'PropBosq'],
       dtype='object'),
 Index(['PorcMujeres', 'PropBosq', 'CrecPobl', 'ConsAgua', 'EmisCO2',
        'ConsEner', 'PNB95', 'ProdElec', 'PropDefor', 'MortInf',
'LinTelf'],
       dtype='object'))
```

Aqui tenemos las variables que afectan a los 2 primeros componentes:

Componente 1: - LinTelf - ConsEner - MortInf - EmisCO2 - CrecPobl

Componente 2: - PorcMujeres - ProBosq - CrecPobl - ConsAgua - EmisCO2

1. Compare los resultados de los incisos 6 y 7. ¿qué concluye?

Varianza explicada:

- Matriz de covarianza: las 2 primeras componentes explican casi el 100% de la varianza.
- Matriz de correlaciones: las 3 primers componentes explican aproximadamente el 66.64% de la varianza.

Variables contribuyentes:

- Matriz de covarianza: las variables PNB95, ProdElec, ConsEner, LinTelf y ConsAgua son especialmente influyentes en los 2 primeros componentes.
- Matriz de correlaciones: las variables: LinTelf, ConsEner, MortInf, EmisCO2, CrecPobl y PNB95 son las mas importantes en el primer componente, mientras que PorcMujeres, PropBosq, CrecPobl y ConsAgua lo son del segundo componente.

Sensibilidad a escala de variables:

- Matriz de covarianza: sensible a las escalas de las variables. Las variables con mayor varianza o unidades mas grandes podrian dominar los primeros componentes.
- Matriz de correlaciones: No es sensible a las escalas ya que las variables estan estandarizadas.

Conclusion: Si las variables tienen unidades y escalas muy diferentes, como es el caso de muchos indicadores económicos y sociales, el PCA basado en la matriz de correlaciones puede ser más adecuado, ya que no permite que las variables con grandes magnitudes dominen las componentes principales. Sin embargo, si el objetivo es mantener la estructura original de los datos y no se quiere perder información debido a la estandarización, entonces el PCA basado en la matriz de covarianza podría ser la elección correcta.

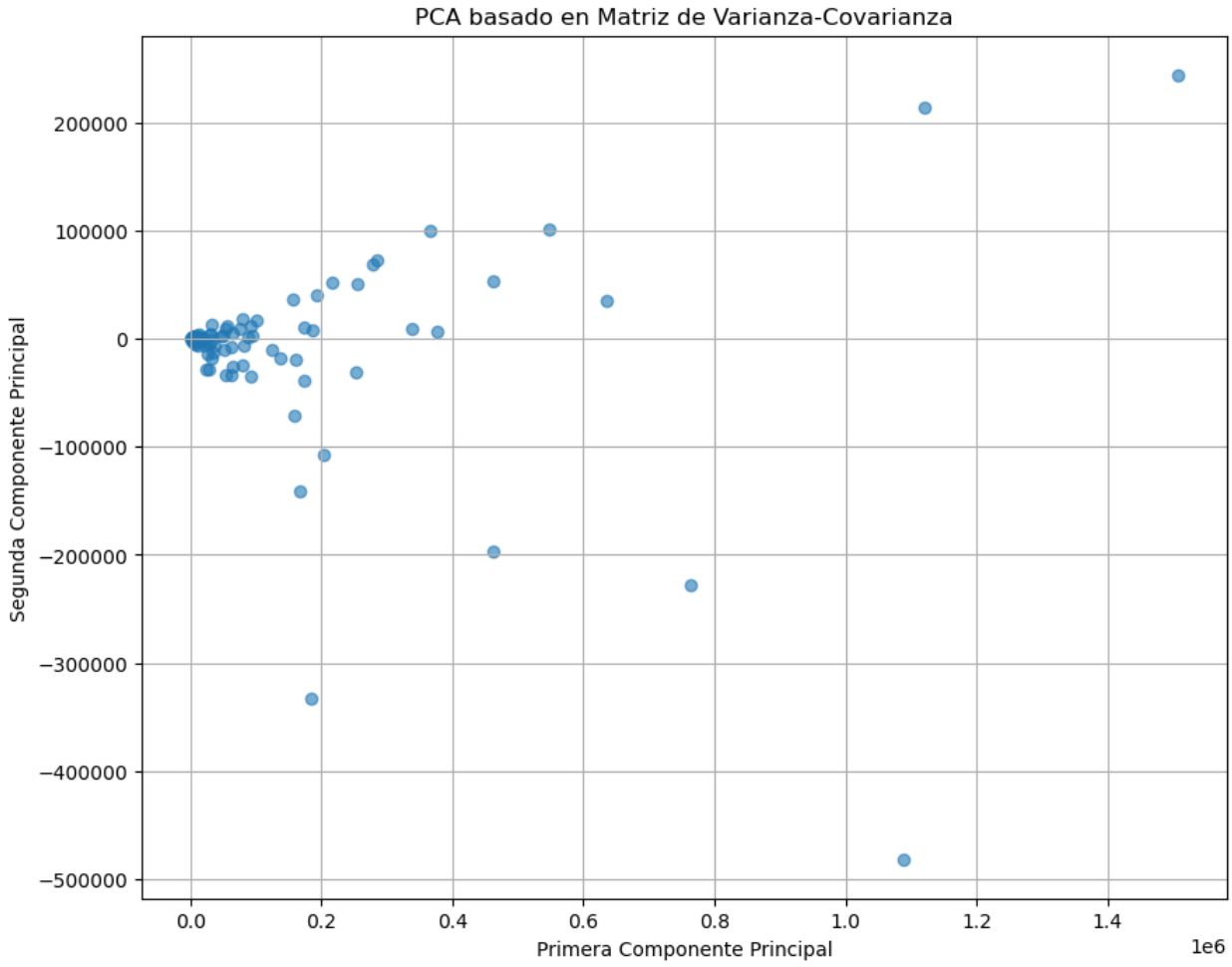
Parte 2

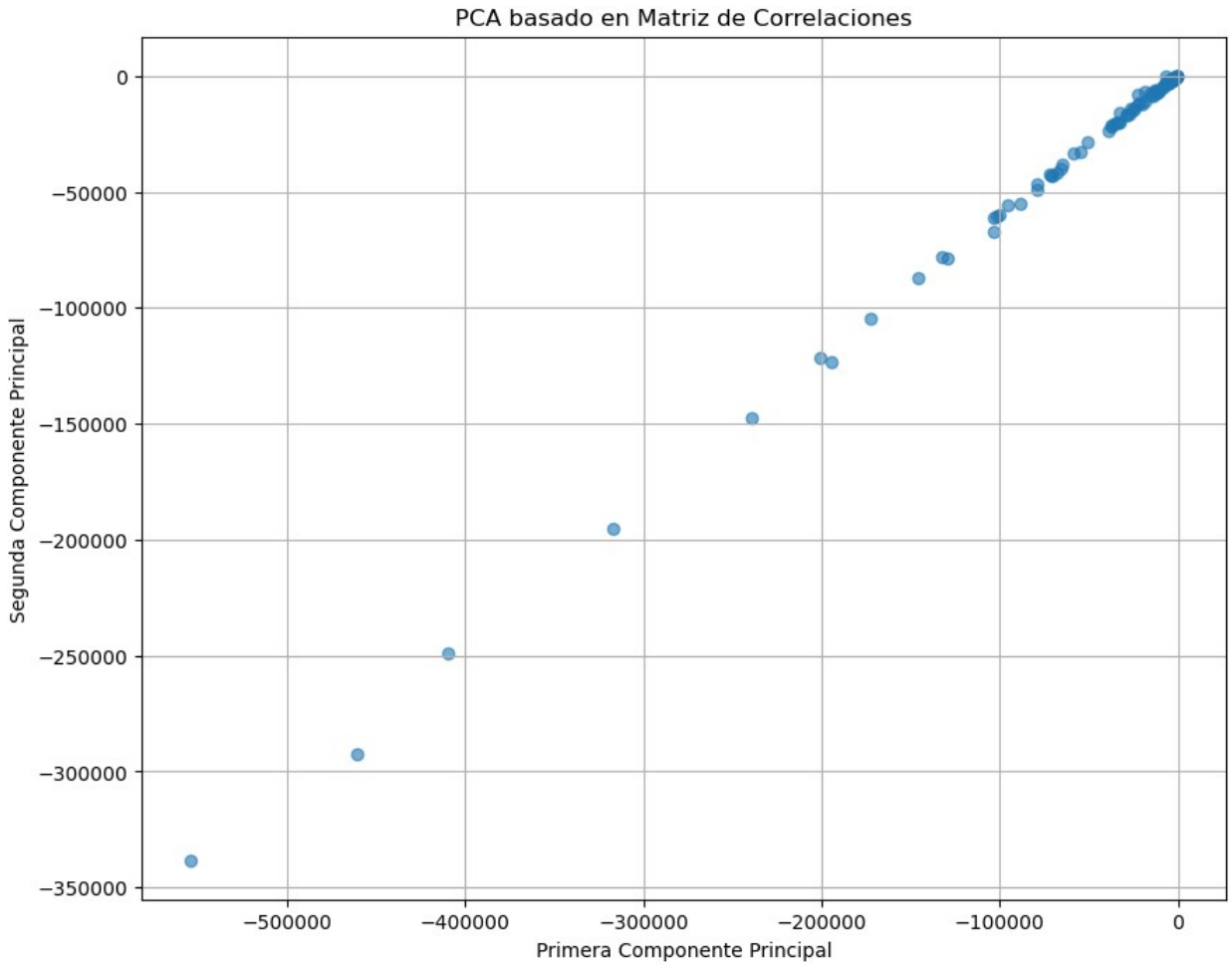
```
scores_cov = df_numerico.dot(eigen_cov_vectores[:, :2])

scores_cor = df_numerico.dot(eigen_cor_vectores[:, :2])

scores_cov.head(), scores_cor.head()
```

```
(                0             1
 0    3748.148614  -2454.860906
 1    4366.215552   1184.993507
 2  160447.555244 -19429.044770
 3   48758.316807   2853.127201
 4  277588.281378  69435.364579,
                0             1
 0   -1809.318461  -1005.905990
 1   -1554.310705   -948.753318
 2  -64998.207474 -38205.543028
 3  -18727.212915 -11089.819026
 4 -100116.559662 -60052.701327)
```

```
def plot_scores_without_labels(scores, title):
    plt.figure(figsize=(10, 8))
    plt.scatter(scores.iloc[:, 0], scores.iloc[:, 1], alpha=0.6)
    plt.title(title)
    plt.xlabel('Primera Componente Principal')
    plt.ylabel('Segunda Componente Principal')
    plt.grid(True)
    plt.show()
```

```
plot_scores_without_labels(scores_cov, "PCA basado en Matriz de
Varianza-Covarianza")
plot_scores_without_labels(scores_cor, "PCA basado en Matriz de
Correlaciones")
```





Interpretacion:

1. PCA basado en Matriz de Varianza-Covarianza:

- La mayoría de los puntos se agrupan cerca del origen, pero hay algunos puntos que se alejan significativamente en la dirección de la primera componente principal.
- Estos puntos que se alejan podrían corresponder a países con valores extremadamente altos en algunas variables, posiblemente indicadores económicos.

1. PCA basado en Matriz de Correlaciones:

- La distribución de los puntos es más equilibrada y centrada alrededor del origen.
- Esto es coherente con lo que esperaríamos, ya que el PCA basado en correlaciones estandariza las variables, lo que significa que todas tienen igual importancia en el análisis.

Parte 3

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

```
scaler = StandardScaler()
data_scaled = scaler.fit_transform(df_numerico)

pca = PCA()
principal_components = pca.fit_transform(data_scaled)

def plot_individuals(principal_components, explained_variance_ratio):
    plt.figure(figsize=(10, 8))
    sns.scatterplot(x=principal_components[:, 0],
y=principal_components[:, 1], alpha=0.6)
    plt.title("Gráfico de las puntuaciones de las componentes
principales")
    plt.xlabel(f"PC1 ({explained_variance_ratio[0]*100:.2f}%)")
    plt.ylabel(f"PC2 ({explained_variance_ratio[1]*100:.2f}%)")
    plt.grid(True)
    plt.show()

def scree_plot(pca):
    plt.figure(figsize=(10, 6))
    plt.plot(range(1, len(pca.explained_variance_ratio_) + 1),
pca.explained_variance_ratio_, 'o-', linewidth=2)
    plt.title("Gráfico de sedimentación (Scree Plot)")
    plt.xlabel("Componente principal")
    plt.ylabel("Varianza explicada")
    plt.grid(True)
    plt.show()

def variable_contribution(pca, df):
    n_pcs = pca.components_.shape[0]
    most_important = [np.abs(pca.components_[i]).argmax() for i in
range(n_pcs)]
    initial_feature_names = df.columns
    most_important_names = [initial_feature_names[most_important[i]]
for i in range(n_pcs)]

    plt.figure(figsize=(12, 6))
    plt.bar(most_important_names, pca.explained_variance_ratio_)
    plt.title("Contribución de cada variable a las componentes
principales")
    plt.ylabel("Varianza explicada")
    plt.show()

plot_individuals(principal_components, pca.explained_variance_ratio_)
scree_plot(pca)
variable_contribution(pca, df_numerico)
```

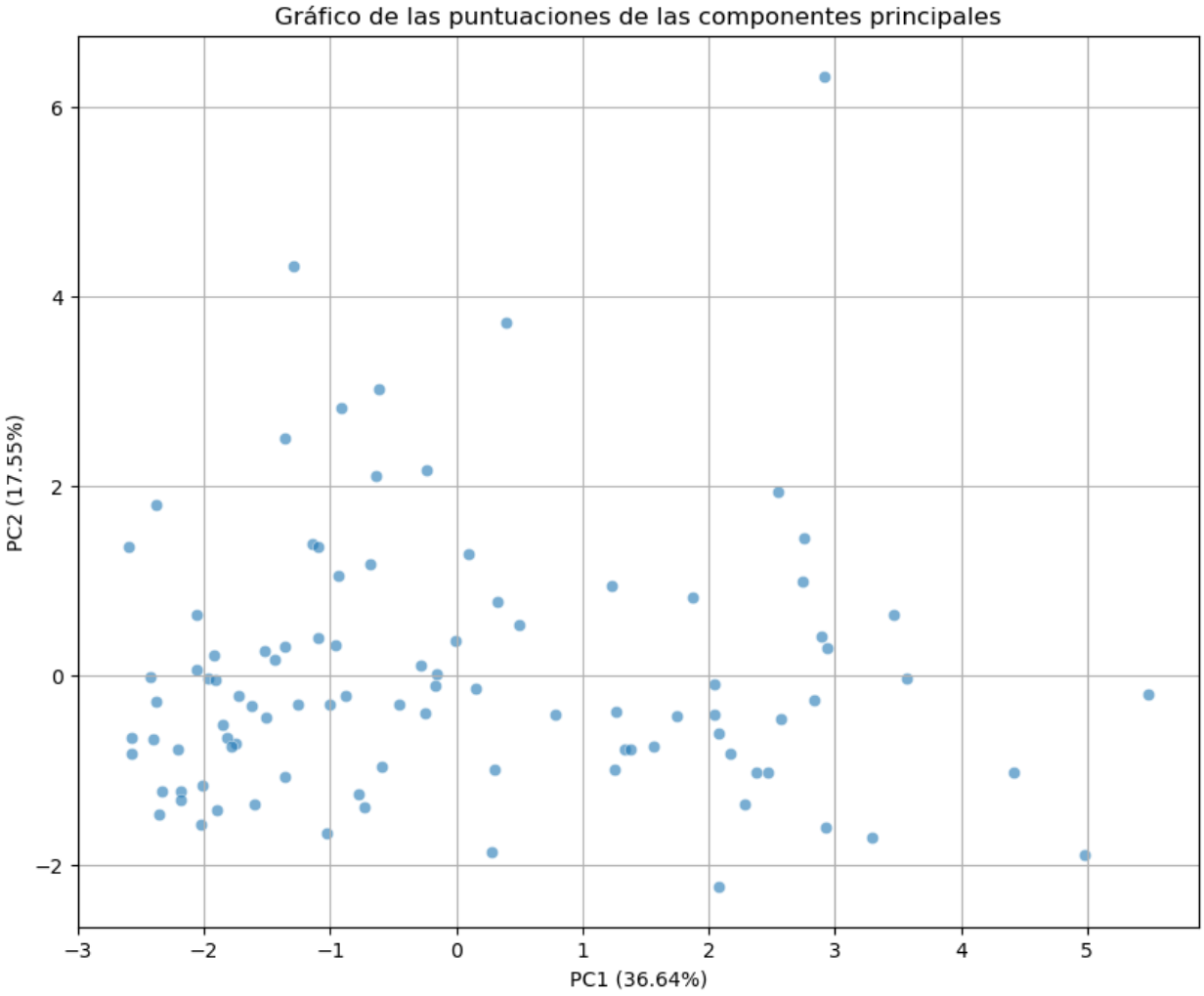
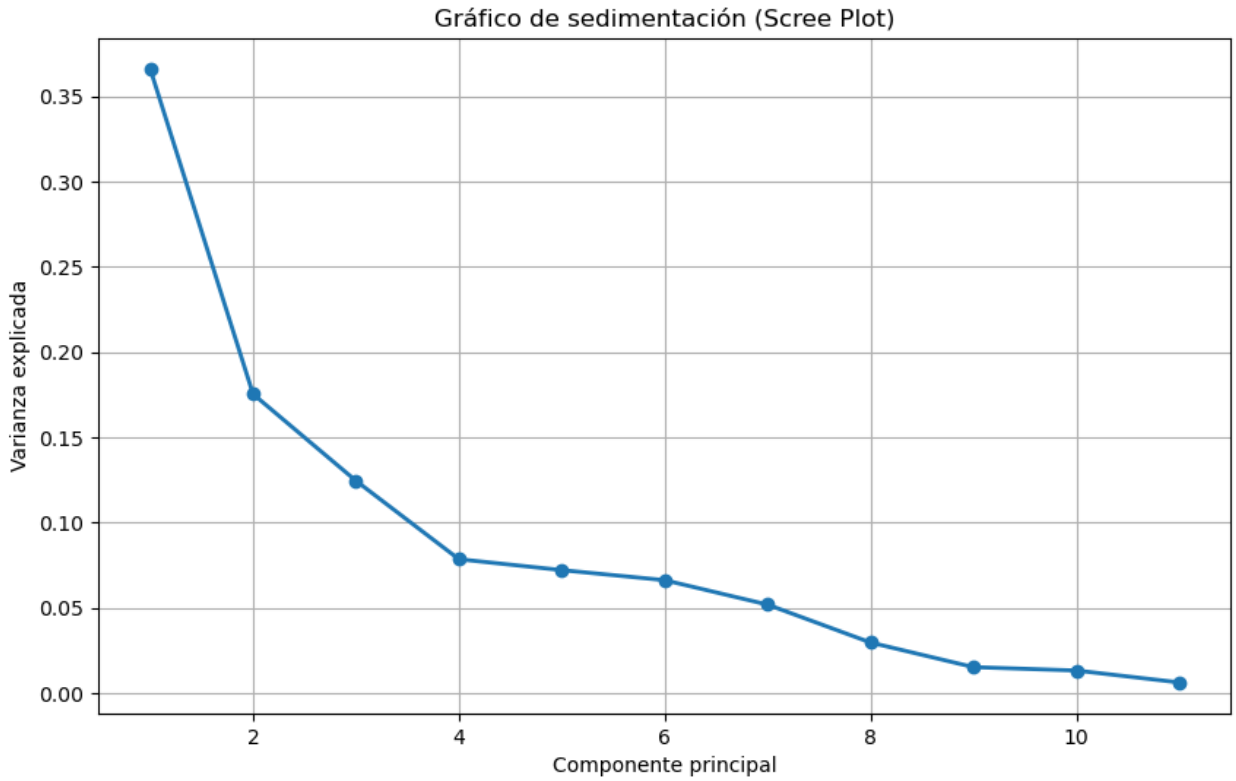
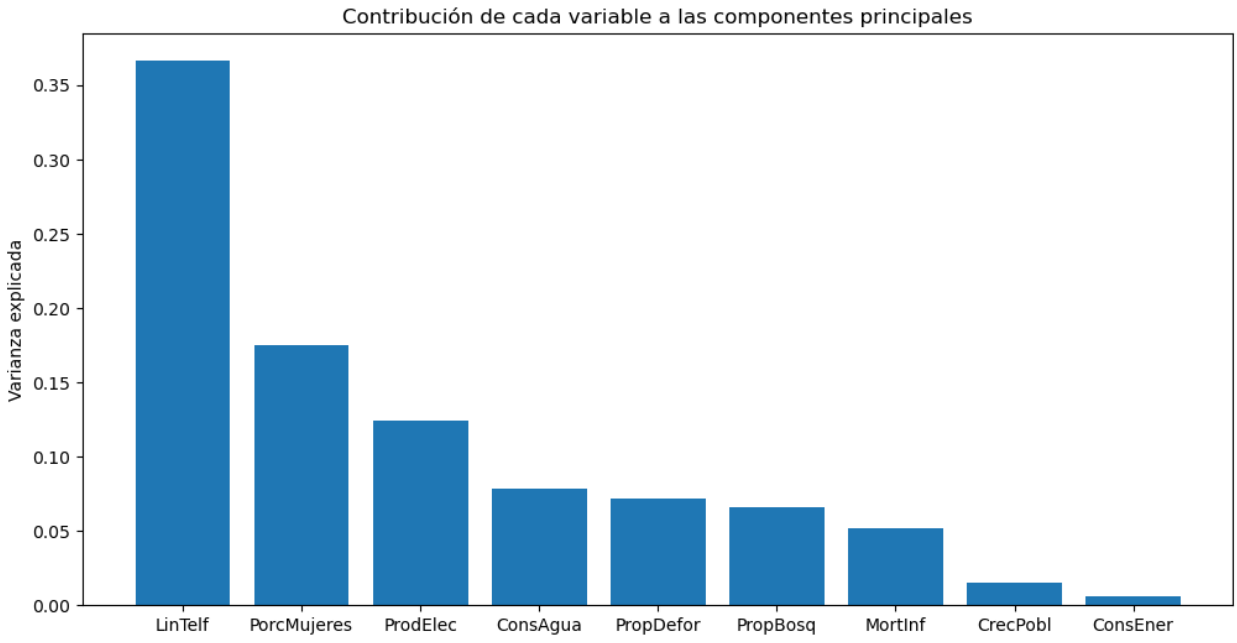


Gráfico de las puntuaciones de las componentes principales
PC2 (17.55%)
PC1 (36.64%)
6
4
2
0
−2
−3
−2
−1
0
1
2
3
4
5





1. Gráfico de las puntuaciones de las componentes principales para los individuos:

- Cada punto representa un país en el espacio de las dos primeras componentes principales.
- Las etiquetas en los ejes x e y indican la proporción de varianza explicada por cada componente principal.

1. Gráfico de sedimentación (Scree Plot):

- Muestra la proporción de varianza explicada por cada componente principal.
- Es útil para determinar cuántas componentes deberías considerar en tu análisis.

1. Contribución de cada variable a las componentes principales:

- Muestra qué variable tiene la mayor contribución (en términos absolutos) a cada componente principal.
- Las barras representan la varianza explicada por cada componente principal, y las etiquetas en el eje x indican qué variable contribuye más a esa componente.