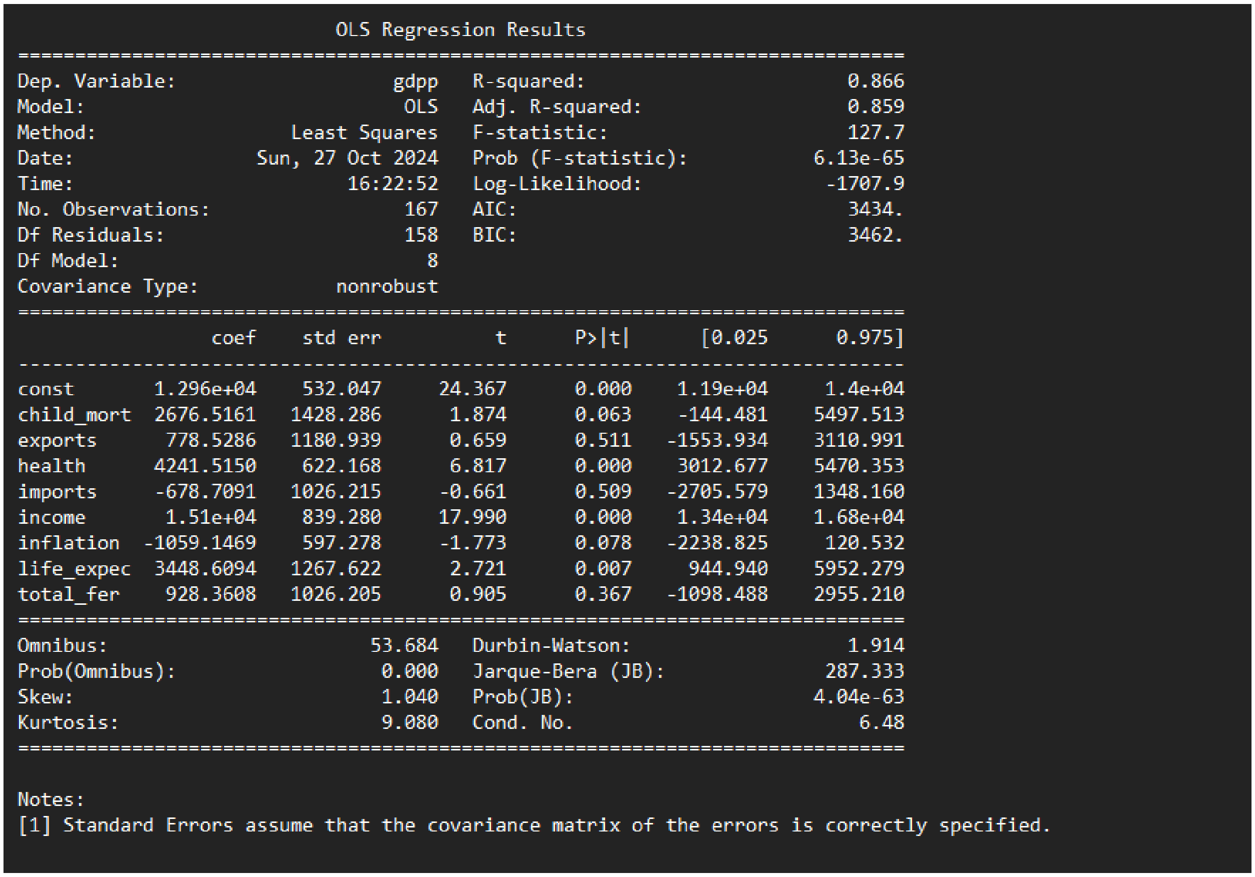
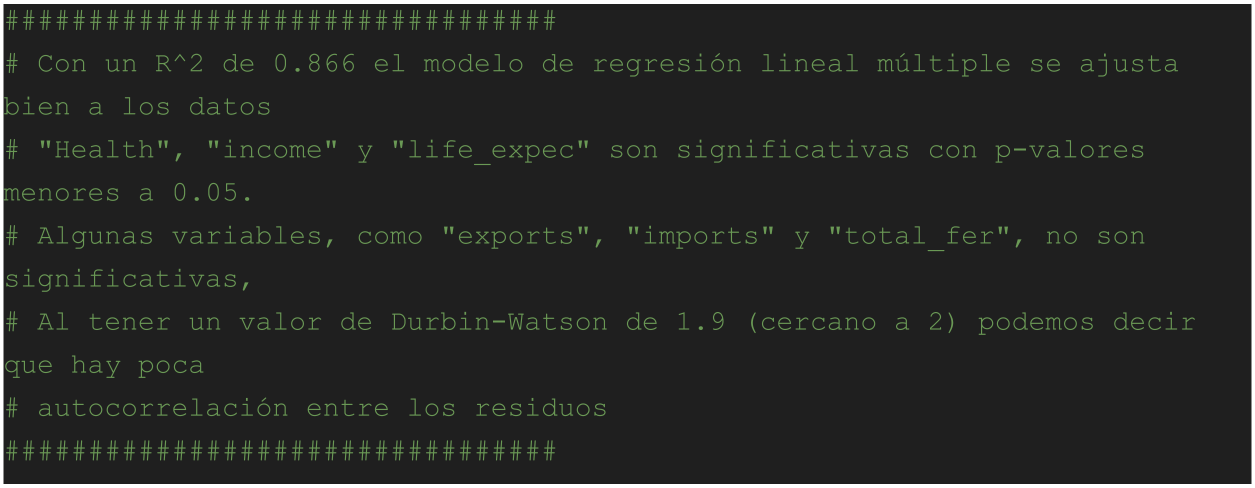
Componentes Principales

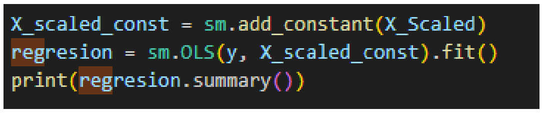
Regresión Lineal:



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   gdpp   R-squared:                       0.866
Model:                            OLS   Adj. R-squared:                  0.859
Method:                 Least Squares   F-statistic:                     127.7
Date:                Sun, 27 Oct 2024   Prob (F-statistic):           6.13e-65
Time:                        16:22:52   Log-Likelihood:                -1707.9
No. Observations:                 167   AIC:                             3434.
Df Residuals:                     158   BIC:                             3462.
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.296e+04    532.047     24.367      0.000    1.19e+04     1.4e+04
child_mort  2676.5161   1428.286      1.874      0.063    -144.481    5497.513
exports      778.5286   1180.939      0.659      0.511   -1553.934    3110.991
health      4241.5150    622.168      6.817      0.000    3012.677    5470.353
imports     -678.7091   1026.215     -0.661      0.509   -2705.579    1348.160
income       1.51e+04    839.280     17.990      0.000    1.34e+04    1.68e+04
inflation  -1059.1469    597.278     -1.773      0.078   -2238.825     120.532
life_expec  3448.6094   1267.622      2.721      0.007     944.940    5952.279
total_fer    928.3608   1026.205      0.905      0.367   -1098.488    2955.210
==============================================================================
Omnibus:                       53.684   Durbin-Watson:                   1.914
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              287.333
Skew:                           1.040   Prob(JB):                     4.04e-63
Kurtosis:                       9.080   Cond. No.                         6.48
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

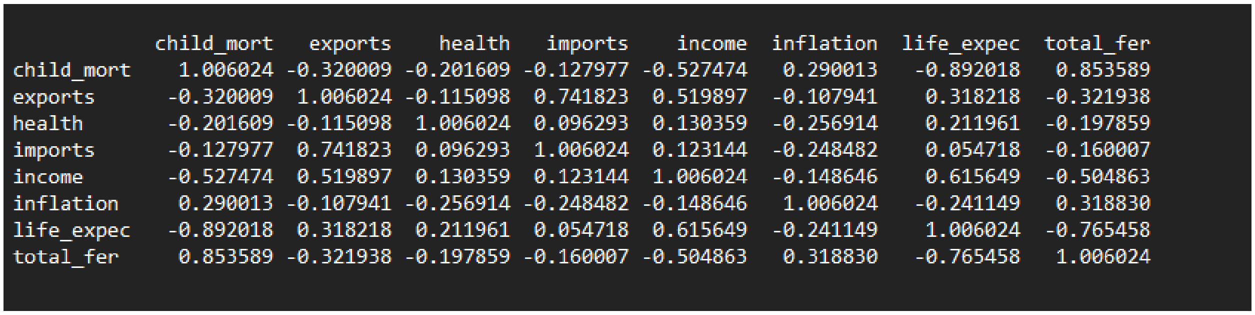


```
##################################
# Con un R^2 de 0.866 el modelo de regresión lineal múltiple se ajusta
bien a los datos
# "Health", "income" y "life_expec" son significativas con p-valores
menores a 0.05.
# Algunas variables, como "exports", "imports" y "total_fer", no son
significativas,
# Al tener un valor de Durbin-Watson de 1.9 (cercano a 2) podemos decir
que hay poca
# autocorrelación entre los residuos
##################################
```

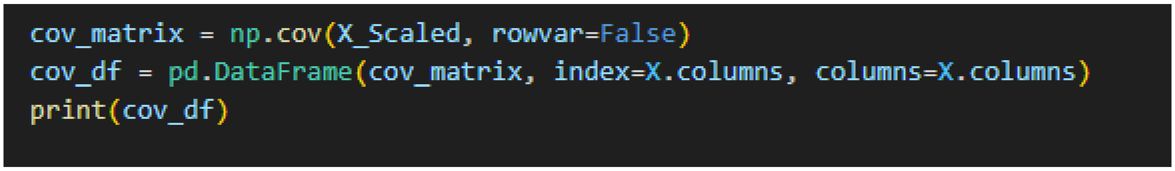


```
X_scaled_const = sm.add_constant(X_Scaled)
regresion = sm.OLS(y, X_scaled_const).fit()
print(regresion.summary())
```

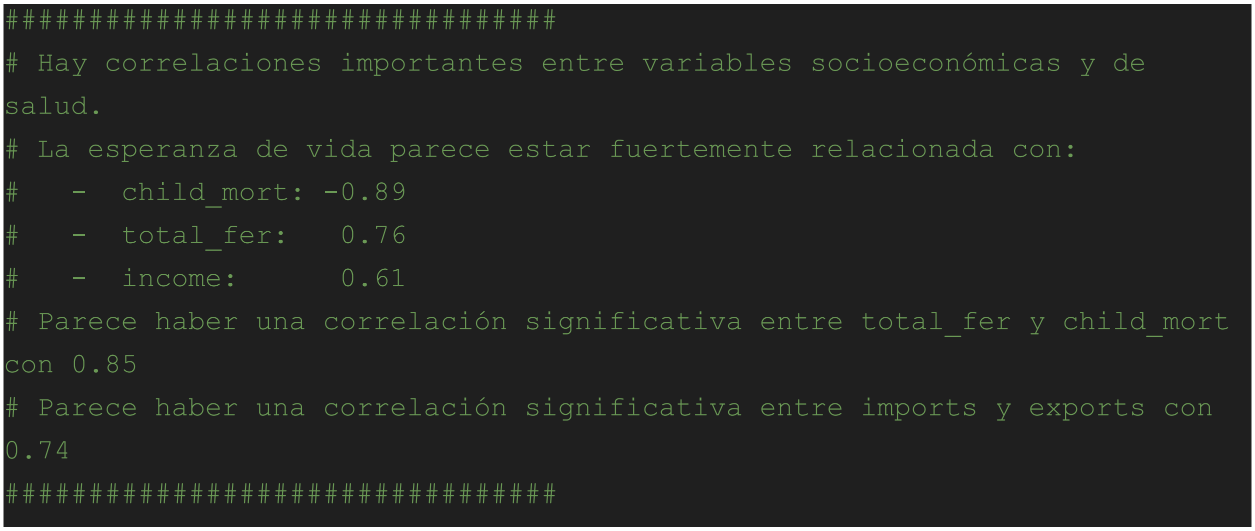
Matriz de covarianza:



```
            child_mort   exports    health   imports    income  inflation  life_expec  total_fer
child_mort    1.006024 -0.320009 -0.201609 -0.127977 -0.527474   0.290013   -0.892018   0.853589
exports      -0.320009  1.006024 -0.115098  0.741823  0.519897  -0.107941    0.318218  -0.321938
health       -0.201609 -0.115098  1.006024  0.096293  0.130359  -0.256914    0.211961  -0.197859
imports      -0.127977  0.741823  0.096293  1.006024  0.123144  -0.248482    0.054718  -0.160007
income       -0.527474  0.519897  0.130359  0.123144  1.006024  -0.148646    0.615649  -0.504863
inflation     0.290013 -0.107941 -0.256914 -0.248482 -0.148646   1.006024   -0.241149   0.318830
life_expec   -0.892018  0.318218  0.211961  0.054718  0.615649  -0.241149    1.006024  -0.765458
total_fer     0.853589 -0.321938 -0.197859 -0.160007 -0.504863   0.318830   -0.765458   1.006024
```

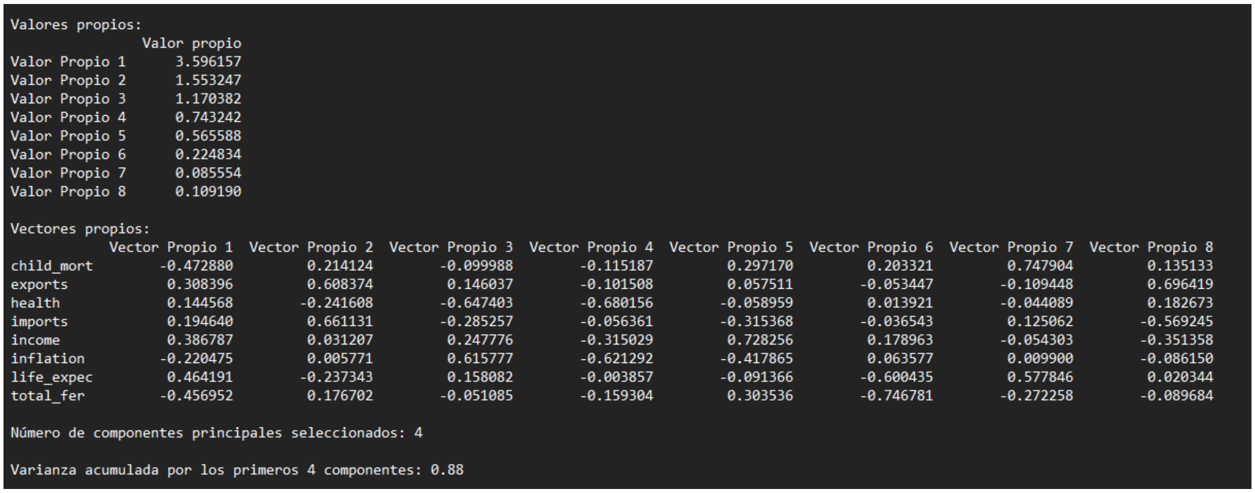


```python
cov_matrix = np.cov(X_Scaled, rowvar=False)
cov_df = pd.DataFrame(cov_matrix, index=X.columns, columns=X.columns)
print(cov_df)
```

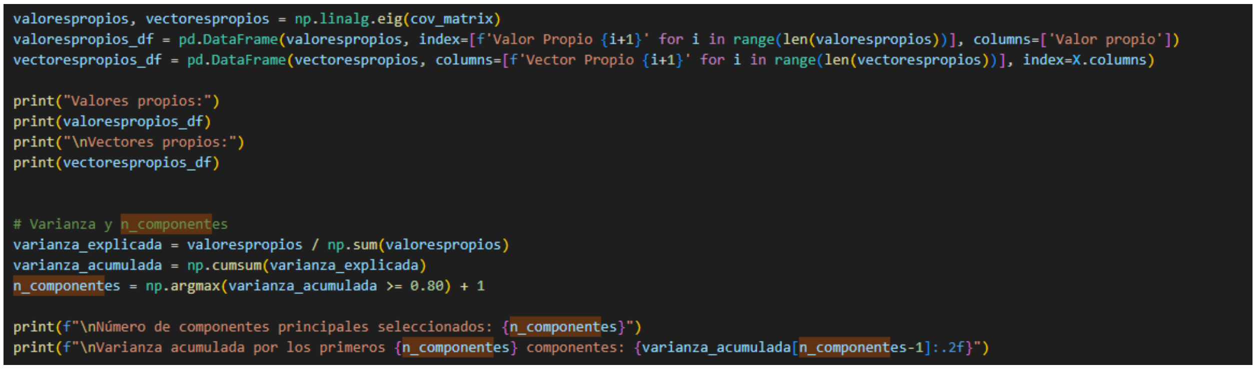


```
#################################
# Hay correlaciones importantes entre variables socioeconómicas y de
salud.
# La esperanza de vida parece estar fuertemente relacionada con:
#   -  child_mort: -0.89
#   -  total_fer:   0.76
#   -  income:      0.61
# Parece haber una correlación significativa entre total_fer y child_mort
con 0.85
# Parece haber una correlación significativa entre imports y exports con
0.74
#################################
```

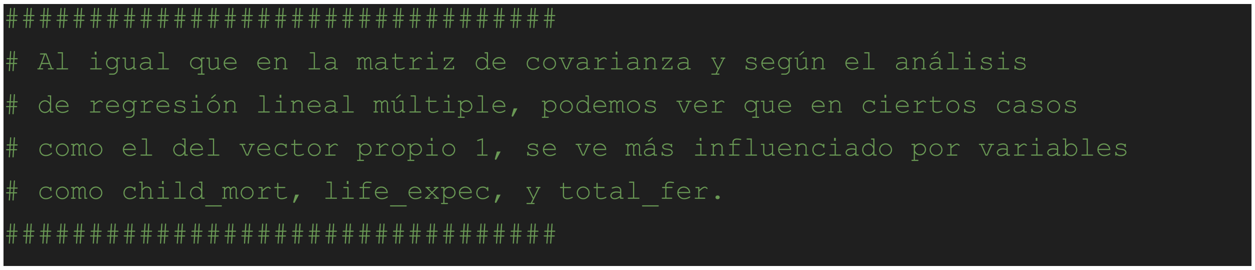
Valores y vectores Propios:



```
Valores propios:
                Valor propio
Valor Propio 1      3.596157
Valor Propio 2      1.553247
Valor Propio 3      1.170382
Valor Propio 4      0.743242
Valor Propio 5      0.565588
Valor Propio 6      0.224834
Valor Propio 7      0.085554
Valor Propio 8      0.109190

Vectores propios:
            Vector Propio 1  Vector Propio 2  Vector Propio 3  Vector Propio 4  Vector Propio 5  Vector Propio 6  Vector Propio 7  Vector Propio 8
child_mort        -0.472880         0.214124        -0.099988        -0.115187         0.297170         0.203321         0.747904         0.135133
exports            0.308396         0.608374         0.146037        -0.101508         0.057511        -0.053447        -0.109448         0.696419
health             0.144568        -0.241608        -0.647403        -0.680156        -0.058959         0.013921        -0.044089         0.182673
imports            0.194640         0.661131        -0.285257        -0.056361        -0.315368        -0.036543         0.125062        -0.569245
income             0.386787         0.031207         0.247776        -0.315029         0.728256         0.178963        -0.054303        -0.351358
inflation         -0.220475         0.005771         0.615777        -0.621292        -0.417865         0.063577         0.009900        -0.086150
life_expec         0.464191        -0.237343         0.158082        -0.003857        -0.091366        -0.600435         0.577846         0.020344
total_fer         -0.456952         0.176702        -0.051085        -0.159304         0.303536        -0.746781        -0.272258        -0.089684

Número de componentes principales seleccionados: 4

Varianza acumulada por los primeros 4 componentes: 0.88
```

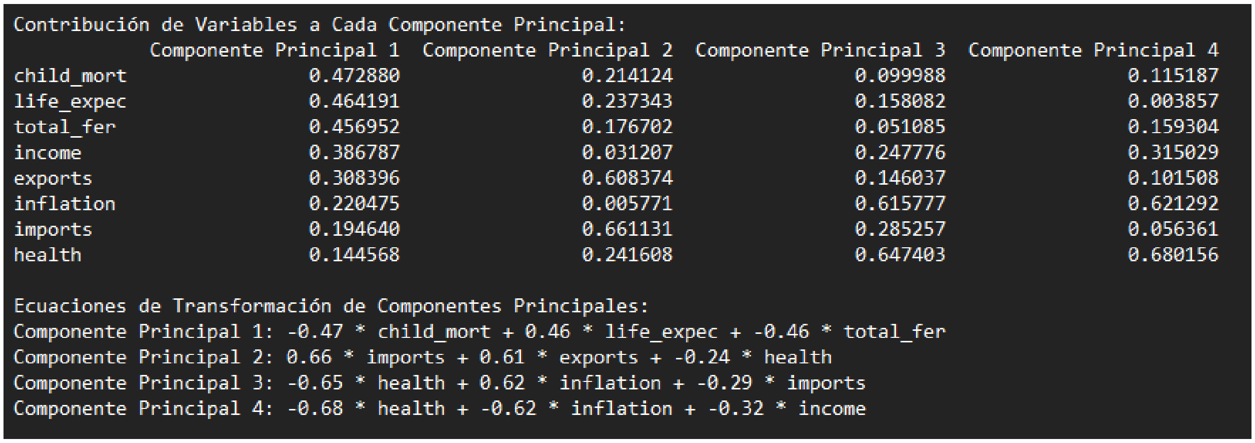


```python
valorespropios, vectorespropios = np.linalg.eig(cov_matrix)
valorespropios_df = pd.DataFrame(valorespropios, index=[f'Valor Propio {i+1}' for i in range(len(valorespropios))], columns=['Valor propio'])
vectorespropios_df = pd.DataFrame(vectorespropios, columns=[f'Vector Propio {i+1}' for i in range(len(vectorespropios))], index=X.columns)

print("Valores propios:")
print(valorespropios_df)
print("\nVectores propios:")
print(vectorespropios_df)


# Varianza y n_componentes
varianza_explicada = valorespropios / np.sum(valorespropios)
varianza_acumulada = np.cumsum(varianza_explicada)
n_componentes = np.argmax(varianza_acumulada >= 0.80) + 1

print(f"\nNúmero de componentes principales seleccionados: {n_componentes}")
print(f"\nVarianza acumulada por los primeros {n_componentes} componentes: {varianza_acumulada[n_componentes-1]:.2f}")
```

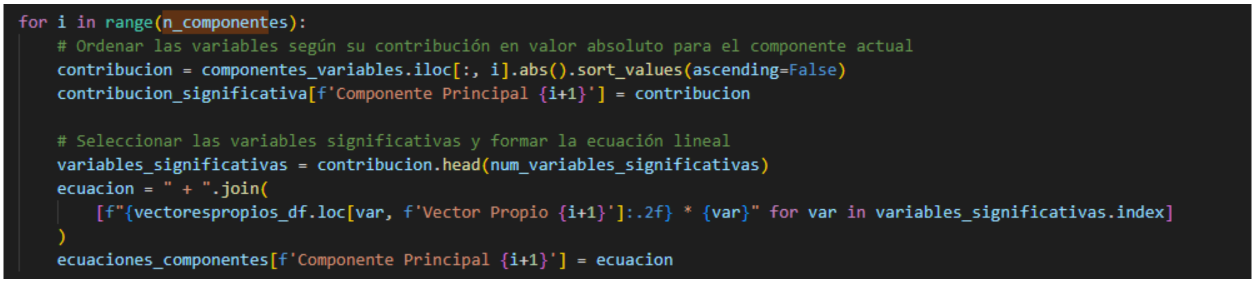


```
#################################
# Al igual que en la matriz de covarianza y según el análisis
# de regresión lineal múltiple, podemos ver que en ciertos casos
# como el del vector propio 1, se ve más influenciado por variables
# como child_mort, life_expec, y total_fer.
#################################
```

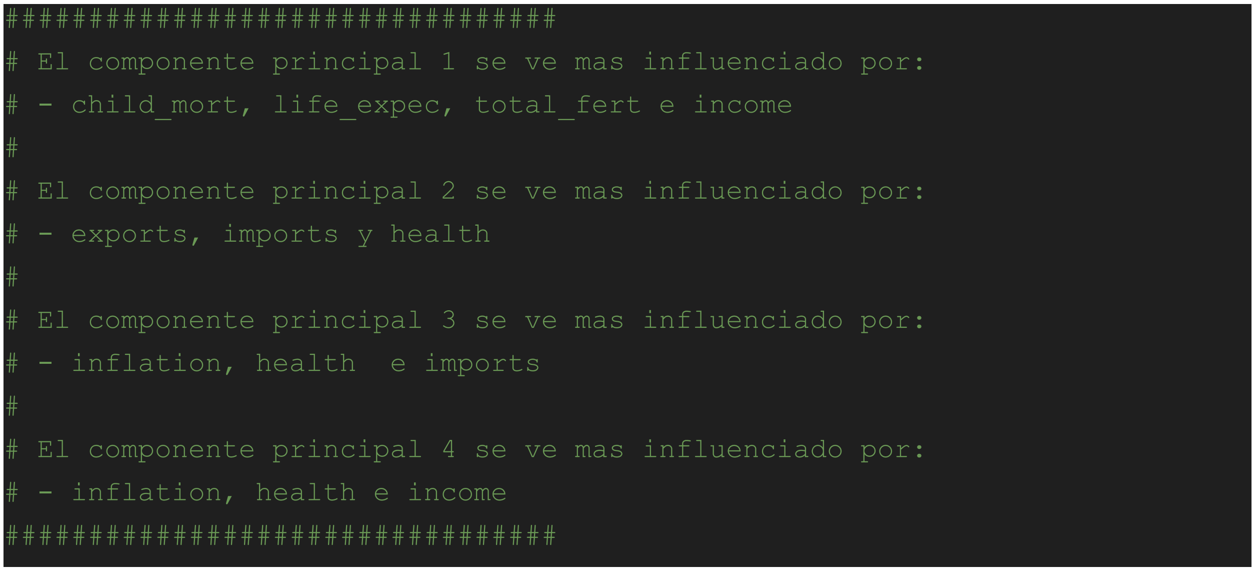
Componentes principales y su ecuación:



```
Contribución de Variables a Cada Componente Principal:
            Componente Principal 1  Componente Principal 2  Componente Principal 3  Componente Principal 4
child_mort                0.472880                0.214124                0.099988                0.115187
life_expec                0.464191                0.237343                0.158082                0.003857
total_fer                 0.456952                0.176702                0.051085                0.159304
income                    0.386787                0.031207                0.247776                0.315029
exports                   0.308396                0.608374                0.146037                0.101508
inflation                 0.220475                0.005771                0.615777                0.621292
imports                   0.194640                0.661131                0.285257                0.056361
health                    0.144568                0.241608                0.647403                0.680156

Ecuaciones de Transformación de Componentes Principales:
Componente Principal 1: -0.47 * child_mort + 0.46 * life_expec + -0.46 * total_fer
Componente Principal 2: 0.66 * imports + 0.61 * exports + -0.24 * health
Componente Principal 3: -0.65 * health + 0.62 * inflation + -0.29 * imports
Componente Principal 4: -0.68 * health + -0.62 * inflation + -0.32 * income
```

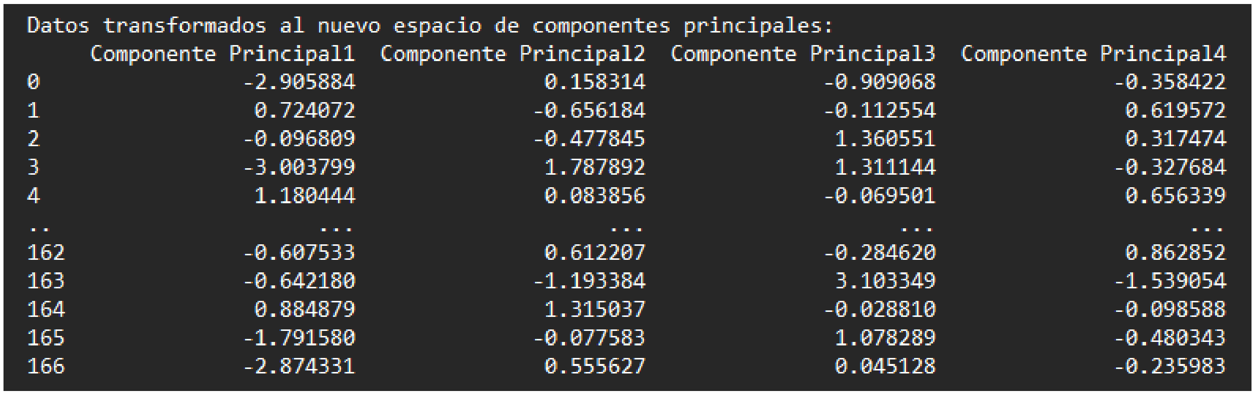


```python
for i in range(n_componentes):
    # Ordenar las variables según su contribución en valor absoluto para el componente actual
    contribucion = componentes_variables.iloc[:, i].abs().sort_values(ascending=False)
    contribucion_significativa[f'Componente Principal {i+1}'] = contribucion

    # Seleccionar las variables significativas y formar la ecuación lineal
    variables_significativas = contribucion.head(num_variables_significativas)
    ecuacion = " + ".join(
        [f"{vectorespropios_df.loc[var, f'Vector Propio {i+1}']:.2f} * {var}" for var in variables_significativas.index]
    )
    ecuaciones_componentes[f'Componente Principal {i+1}'] = ecuacion
```

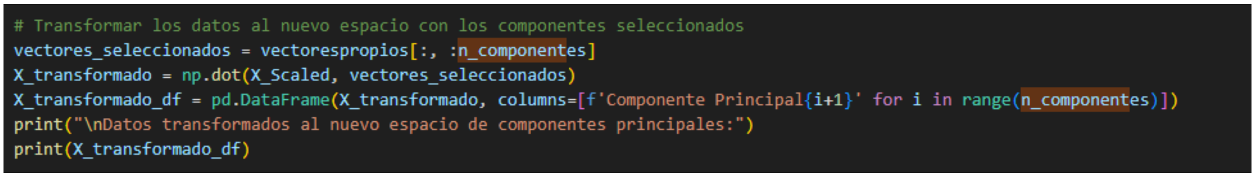


```
#################################
# El componente principal 1 se ve mas influenciado por:
# - child_mort, life_expec, total_fert e income
#
# El componente principal 2 se ve mas influenciado por:
# - exports, imports y health
#
# El componente principal 3 se ve mas influenciado por:
# - inflation, health  e imports
#
# El componente principal 4 se ve mas influenciado por:
# - inflation, health e income
#################################
```

Datos Transformados

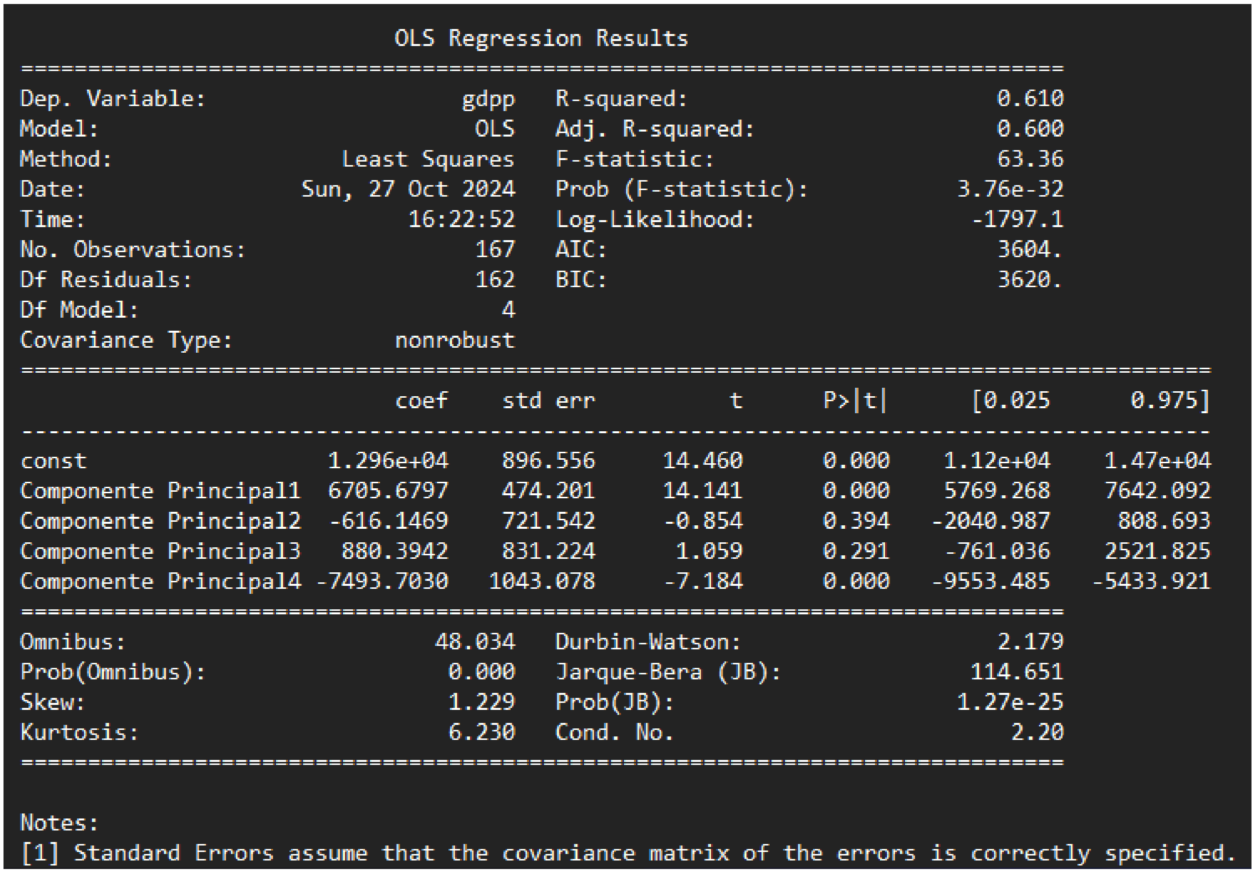


```
Datos transformados al nuevo espacio de componentes principales:
     Componente Principal1  Componente Principal2  Componente Principal3  Componente Principal4
0                -2.905884               0.158314              -0.909068              -0.358422
1                 0.724072              -0.656184              -0.112554               0.619572
2                -0.096809              -0.477845               1.360551               0.317474
3                -3.003799               1.787892               1.311144              -0.327684
4                 1.180444               0.083856              -0.069501               0.656339
..                     ...                    ...                    ...                    ...
162              -0.607533               0.612207              -0.284620               0.862852
163              -0.642180              -1.193384               3.103349              -1.539054
164               0.884879               1.315037              -0.028810              -0.098588
165              -1.791580              -0.077583               1.078289              -0.480343
166              -2.874331               0.555627               0.045128              -0.235983
```

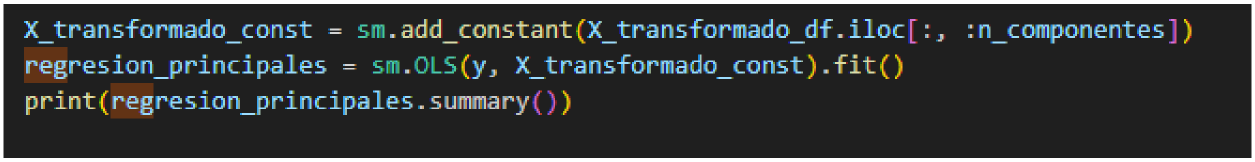


```
# Transformar los datos al nuevo espacio con los componentes seleccionados
vectores_seleccionados = vectorespropios[:, :n_componentes]
X_transformado = np.dot(X_Scaled, vectores_seleccionados)
X_transformado_df = pd.DataFrame(X_transformado, columns=[f'Componente Principal{i+1}' for i in range(n_componentes)])
print("\nDatos transformados al nuevo espacio de componentes principales:")
print(X_transformado_df)
```

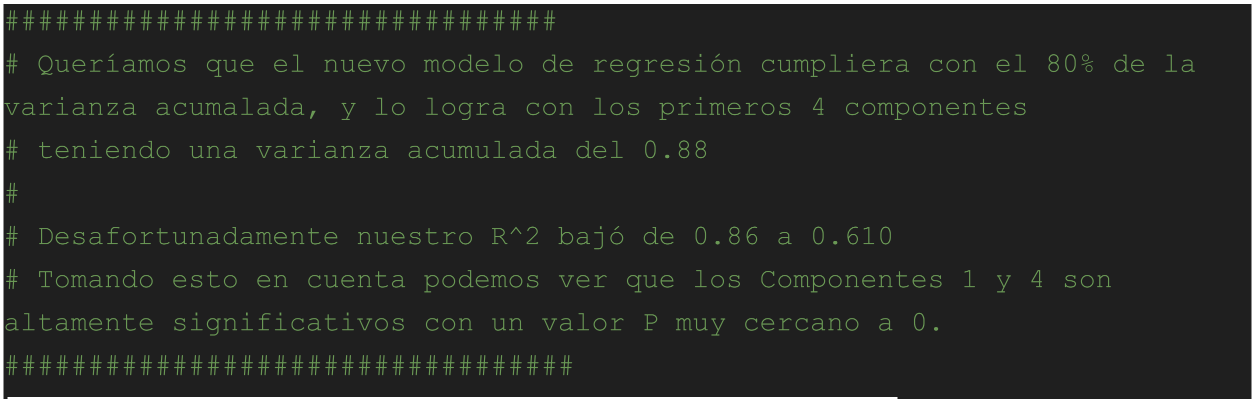
Nueva Regresión Lineal (Con Componentes Principales):



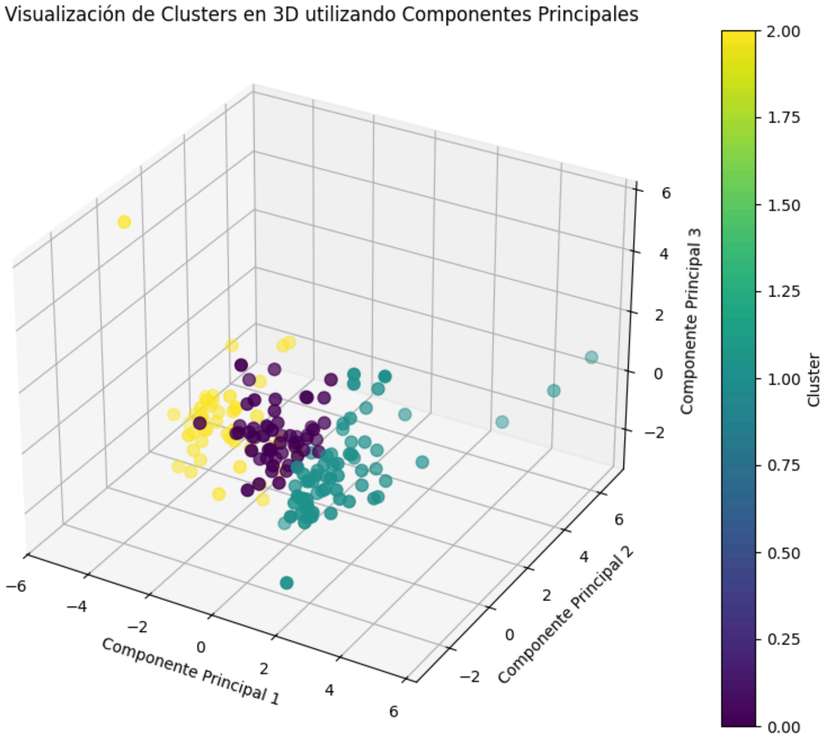
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   gdpp   R-squared:                       0.610
Model:                            OLS   Adj. R-squared:                  0.600
Method:                 Least Squares   F-statistic:                     63.36
Date:                Sun, 27 Oct 2024   Prob (F-statistic):           3.76e-32
Time:                        16:22:52   Log-Likelihood:                -1797.1
No. Observations:                 167   AIC:                             3604.
Df Residuals:                     162   BIC:                             3620.
Df Model:                           4
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
const                  1.296e+04    896.556     14.460      0.000    1.12e+04    1.47e+04
Componente Principal1  6705.6797    474.201     14.141      0.000    5769.268    7642.092
Componente Principal2  -616.1469    721.542     -0.854      0.394   -2040.987     808.693
Componente Principal3   880.3942    831.224      1.059      0.291    -761.036    2521.825
Componente Principal4 -7493.7030   1043.078     -7.184      0.000   -9553.485   -5433.921
==============================================================================
Omnibus:                       48.034   Durbin-Watson:                   2.179
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              114.651
Skew:                           1.229   Prob(JB):                     1.27e-25
Kurtosis:                       6.230   Cond. No.                         2.20
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```



```
X_transformado_const = sm.add_constant(X_transformado_df.iloc[:, :n_componentes])
regresion_principales = sm.OLS(y, X_transformado_const).fit()
print(regresion_principales.summary())
```



```
##################################
# Queríamos que el nuevo modelo de regresión cumpliera con el 80% de la
varianza acumalada, y lo logra con los primeros 4 componentes
# teniendo una varianza acumulada del 0.88
#
# Desafortunadamente nuestro R^2 bajó de 0.86 a 0.610
# Tomando esto en cuenta podemos ver que los Componentes 1 y 4 son
altamente significativos con un valor P muy cercano a 0.
##################################
```


Visualización de Clusters en 3D utilizando Componentes Principales
Componente Principal 1
Componente Principal 2
Componente Principal 3
Cluster