# Tarea_4_Cardenas_Venegas

December 19, 2022

# 1 Tarea#4_Cardenas_Venegas

    diciembre 9, 2022

## SEM Autores : David Càrdenas y Cristobal Venegas

[1]:
```
# Tratamiento de datos
# ==============================================================================
import numpy as np
import pandas as pd
import statsmodels.api as sm

# Gráficos
# ==============================================================================
import matplotlib.pyplot as plt
import matplotlib.font_manager
from matplotlib import style
style.use('ggplot') or plt.style.use('ggplot')
import seaborn as sns

# Preprocesado y modelado
# ==============================================================================
from sklearn.decomposition import PCA
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import scale

# Configuración warnings
# ==============================================================================
import warnings
warnings.filterwarnings('ignore')
```

# 2 Limpieza de datos

[2]:
```
junaeb2 = pd.read_csv("C:/Users/crist/Documents/GitHub/LAB-MAA_1/data/junaeb2.
↪csv")
```

[3]:
```
junaeb2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 59999 entries, 0 to 59998
Data columns (total 23 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   sexo        59999 non-null  int64
 1   edad        59999 non-null  int64
 2   imce        59999 non-null  float64
 3   vive_padre  59999 non-null  int64
 4   vive_madre  59999 non-null  int64
 5   sk1         59999 non-null  int64
 6   sk2         59999 non-null  int64
 7   sk3         59999 non-null  int64
 8   sk4         59999 non-null  int64
 9   sk5         59999 non-null  int64
 10  sk6         59999 non-null  int64
 11  sk7         59999 non-null  int64
 12  sk8         59999 non-null  int64
 13  sk9         59999 non-null  int64
 14  sk10        59999 non-null  int64
 15  sk11        59999 non-null  int64
 16  sk12        59999 non-null  int64
 17  sk13        59999 non-null  int64
 18  act_fisica  58033 non-null  float64
 19  area        59999 non-null  int64
 20  educm       59278 non-null  float64
 21  educp       59999 non-null  int64
 22  madre_work  59999 non-null  int64
dtypes: float64(3), int64(20)
memory usage: 10.5 MB
```

[4]:
```
junaeb2.isnull().sum().sort_values(ascending=False)
```

[4]:
```
act_fisica    1966
educm          721
sexo             0
sk8              0
educp            0
area             0
sk13             0
sk12             0
sk11             0
sk10             0
sk9              0
sk7              0
```

```
edad            0
sk6             0
sk5             0
sk4             0
sk3             0
sk2             0
sk1             0
vive_madre      0
vive_padre      0
imce            0
madre_work      0
dtype: int64
```

[5]:
```
junaeb2.dropna(inplace=True)
```

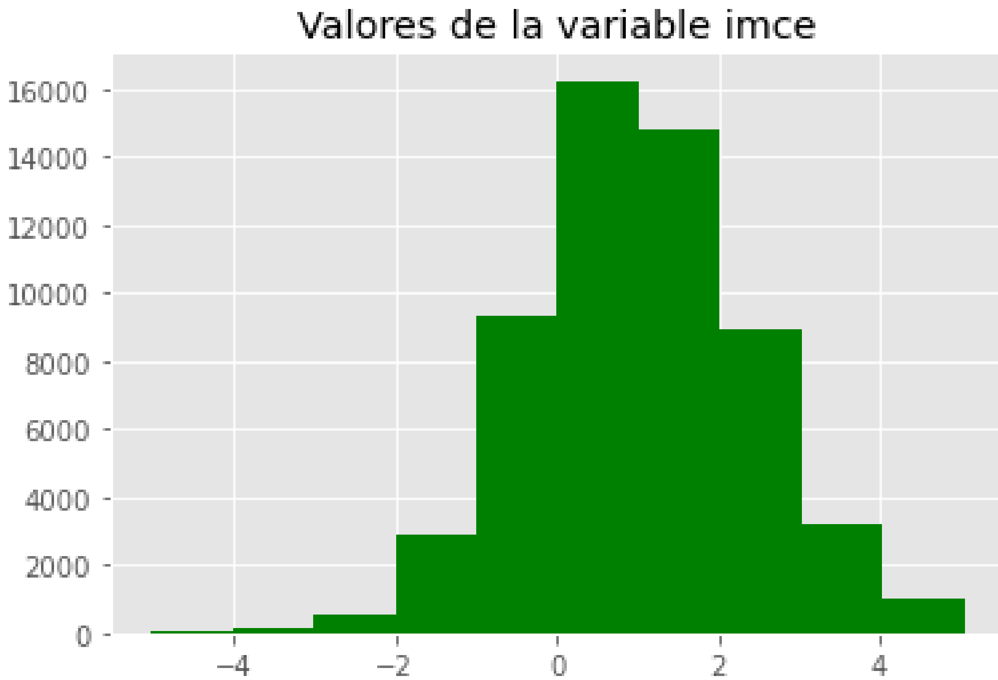
[6]:
```
print ("La variable vive_padre:",junaeb2["vive_padre"].unique(),"\n","La␣
↪variable vive_madre:",junaeb2["vive_madre"].unique())
```

```
La variable vive_padre: [0 1 2]
 La variable vive_madre: [1 0 2]
```

[7]:
```
junaeb2.drop(junaeb2.loc[junaeb2.vive_madre==2].index,inplace=True)
junaeb2.drop(junaeb2.loc[junaeb2.vive_padre==2].index,inplace=True)
```

[8]:
```
plt.hist(junaeb2['imce'],color="green")
plt.title("Valores de la variable imce")
junaeb2.imce.value_counts()
```

[8]:
```
 0.74    208
 1.07    197
 0.87    197
 0.73    195
 0.39    195
        …
-3.35      1
-4.36      1
-4.66      1
-3.17      1
-4.88      1
Name: imce, Length: 928, dtype: int64
```




[9]:
```
junaeb2.drop(junaeb2[junaeb2['imce']<0].index,inplace =True)
junaeb2.reset_index(drop=True, inplace=True)
```

[10]:
```
junaeb2.reset_index(drop=False,inplace=True)
```

[11]:
```
junaeb2.drop(columns=["index"],inplace=True)
```

[12]:
```
junaeb2.describe()
```

[12]:
```
               sexo          edad          imce    vive_padre    vive_madre  \
count  44502.000000  44502.000000  44502.000000  44502.000000  44502.000000
mean       0.536628     81.851759      1.530314      0.719765      0.974338
std        0.498662      3.746846      1.039939      0.449119      0.158126
min        0.000000     62.000000      0.000000      0.000000      0.000000
25%        0.000000     80.000000      0.700000      0.000000      1.000000
50%        1.000000     81.000000      1.360000      1.000000      1.000000
75%        1.000000     83.000000      2.200000      1.000000      1.000000
max        1.000000    107.000000      5.040000      1.000000      1.000000

                sk1           sk2           sk3           sk4           sk5  \
count  44502.000000  44502.000000  44502.000000  44502.000000  44502.000000
mean       1.103950      1.380140      1.252168      1.243135      1.264595
std        0.370878      0.643232      0.570744      0.557179      0.559275
min        1.000000      1.000000      1.000000      1.000000      1.000000
```

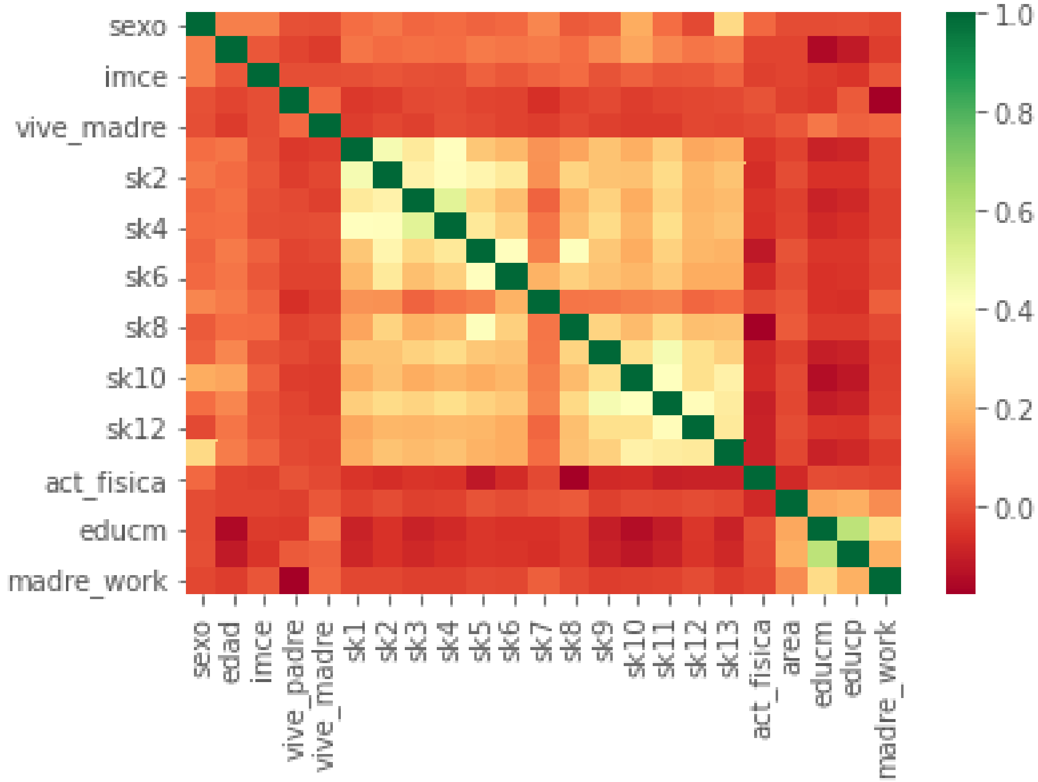
```
25%         1.000000      1.000000      1.000000      1.000000      1.000000
50%         1.000000      1.000000      1.000000      1.000000      1.000000
75%         1.000000      2.000000      1.000000      1.000000      1.000000
max         5.000000      5.000000      5.000000      5.000000      5.000000

       …           sk9          sk10          sk11          sk12  \
count  …  44502.000000  44502.000000  44502.000000  44502.000000
mean   …      1.318862      1.846748      1.372118      1.491708
std    …      0.646841      0.931544      0.652692      0.790304
min    …      1.000000      1.000000      1.000000      1.000000
25%    …      1.000000      1.000000      1.000000      1.000000
50%    …      1.000000      2.000000      1.000000      1.000000
75%    …      1.000000      2.000000      2.000000      2.000000
max    …      5.000000      5.000000      5.000000      5.000000

               sk13    act_fisica          area         educm         educp  \
count  44502.000000  44502.000000  44502.000000  44502.000000  44502.000000
mean       1.684936      2.556537      0.903465     13.066784     12.943890
std        0.978241      1.066804      0.295327      3.319058      3.413612
min        1.000000      1.000000      0.000000      0.000000      0.000000
25%        1.000000      2.000000      1.000000     12.000000     11.000000
50%        1.000000      2.000000      1.000000     13.000000     13.000000
75%        2.000000      3.000000      1.000000     15.000000     14.000000
max        5.000000      5.000000      1.000000     22.000000     22.000000

         madre_work
count  44502.000000
mean       0.107388
std        0.940916
min       -1.000000
25%       -1.000000
50%        1.000000
75%        1.000000
max        1.000000

[8 rows x 23 columns]
```

[13]:

```python
sns.heatmap(junaeb2.corr(), cmap='RdYlGn')
```

[13]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1bea3b4f280>
```



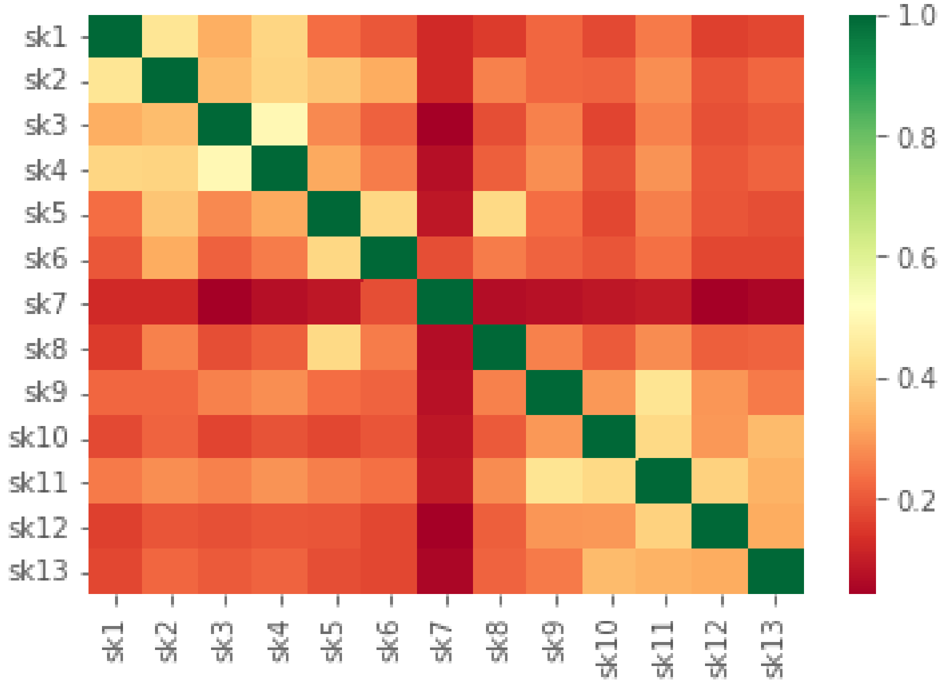

## 3 PCA

[14]:
```
var_sk =␣
↪junaeb2[["sk1","sk2","sk3","sk4","sk5","sk6","sk7","sk8","sk9","sk10","sk11","sk12","sk13"]]
pca = PCA(n_components=12)
pca_features = pca.fit_transform(var_sk)
print(pca.explained_variance_ratio_)
```

```
[0.28606755 0.18817514 0.09973579 0.07929705 0.06632621 0.06471883
 0.05164867 0.0437851  0.03238106 0.0294521  0.02499784 0.02101661]
```

[15]:
```
sns.heatmap(var_sk.corr(), cmap='RdYlGn')
```

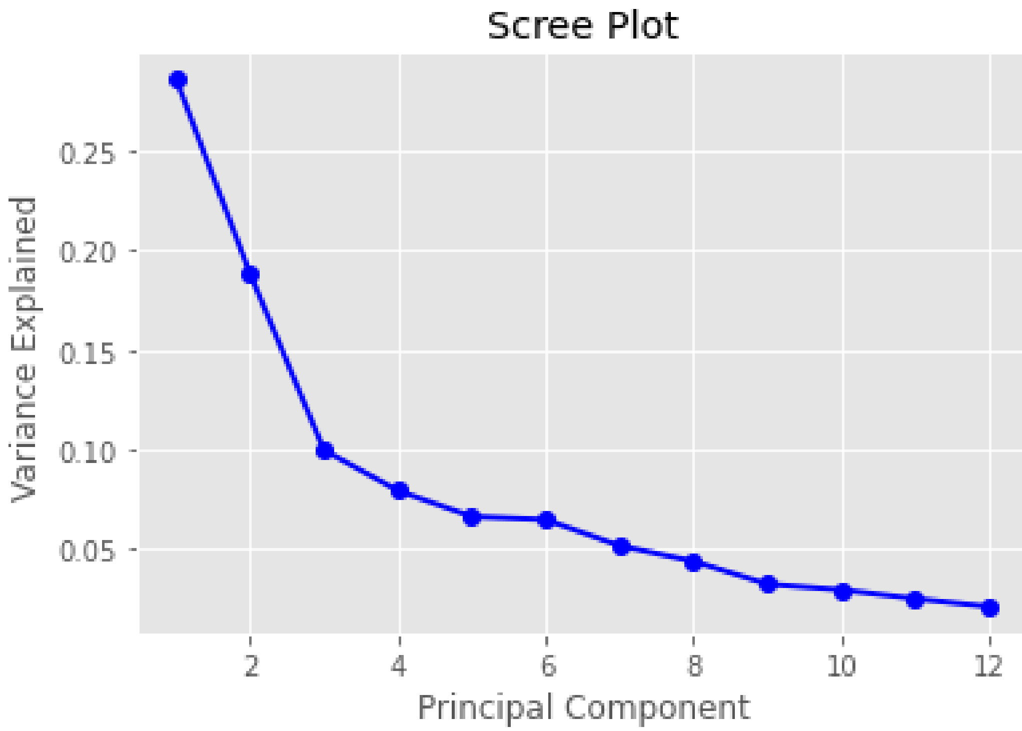
[15]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1bea648c760>
```




[16]:
```
pca.explained_variance_ratio_[0:7].sum()
```

[16]: 0.8359692320811097

[17]:
```
#scree plot using explained variance proportion

PC_values = np.arange(pca.n_components_) + 1
plt.plot(PC_values, pca.explained_variance_ratio_, 'o-', linewidth=2,␣
 ↪color='blue')
plt.title('Scree Plot')
plt.xlabel('Principal Component')
plt.ylabel('Variance Explained')
plt.show()
```




[18]:
```
## Criterio scree plot
pca = PCA(n_components=7)
pca_features = pca.fit_transform(var_sk)
print(pca.explained_variance_ratio_,"\n","Varianza acumulada o explicada por␣
 ↪los primeros 7 componentes:" , pca.explained_variance_ratio_[0:7].sum())
```

```
[0.28606755 0.18817514 0.09973579 0.07929705 0.06632621 0.06471883
 0.05164867]
 Varianza acumulada o explicada por los primeros 7 componentes:
0.8359692320810971
```

[19]:
```
pca_vectors = pd.DataFrame(data = pca.components_,
                           columns=var_sk.columns,
                           index=["PC1","PC2","PC3","PC4","PC5","PC6","PC7"])
pca_vectors
```

[19]:

| | sk1 | sk2 | sk3 | sk4 | sk5 | sk6 | sk7 \ |
|---|---|---|---|---|---|---|---|
| PC1 | 0.101837 | 0.228681 | 0.166704 | 0.180646 | 0.186154 | 0.249874 | 0.351651 |
| PC2 | 0.007632 | 0.034134 | 0.066008 | 0.055333 | 0.040743 | -0.028742 | -0.923058 |
| PC3 | 0.084242 | 0.266514 | 0.180453 | 0.191232 | 0.300941 | 0.341428 | -0.115066 |
| PC4 | -0.004360 | -0.033451 | -0.039866 | -0.031587 | -0.046156 | -0.002758 | -0.032316 |
| PC5 | 0.143585 | 0.268222 | 0.309358 | 0.295992 | -0.021423 | 0.162823 | -0.030117 |
| PC6 | -0.067635 | -0.225324 | -0.119334 | -0.128801 | -0.080588 | -0.270949 | 0.060366 |

```
PC7  0.126372  0.153634  0.323610  0.280628 -0.104650 -0.766114  0.062409

          sk8       sk9      sk10      sk11      sk12      sk13
PC1  0.291481  0.234854  0.406509  0.282787  0.293839  0.429516
PC2  0.099394  0.086327  0.168557  0.104073  0.155922  0.236538
PC3  0.477371  0.079394 -0.373427  0.000937 -0.089114 -0.500784
PC4 -0.095896  0.105624  0.707258  0.138852  0.022035 -0.672875
PC5 -0.746663  0.111445 -0.196699  0.105288  0.259159 -0.096871
PC6  0.170746  0.160812 -0.285903  0.177867  0.786349 -0.208653
PC7  0.137519  0.252963 -0.034518  0.143796 -0.263787 -0.028619
```

[20]:
```python
##Criterio MLE
pca = PCA(n_components='mle')
pca_features = pca.fit_transform(var_sk)
print(pca.explained_variance_ratio_)
```

```
[0.28606755 0.18817514 0.09973579 0.07929705 0.06632621 0.06471883
 0.05164867 0.0437851  0.03238106 0.0294521  0.02499784 0.02101661]
```

A continuación podemos ver los pesos relativos que indican cómo se relaciona cada variable con los factores.

[21]:
```python
pca_vectors = pd.DataFrame(data = pca.components_,
                          columns=var_sk.columns,
                          ␣
 ↪index=["PC1","PC2","PC3","PC4","PC5","PC6","PC7","PC8","PC9","PC10","PC11","PC12"])
pca_vectors
```

[21]:
```
           sk1       sk2       sk3       sk4       sk5       sk6       sk7  \
PC1   0.101837  0.228681  0.166704  0.180646  0.186154  0.249874  0.351651
PC2   0.007632  0.034134  0.066008  0.055333  0.040743 -0.028742 -0.923058
PC3   0.084242  0.266514  0.180453  0.191232  0.300941  0.341428 -0.115066
PC4  -0.004360 -0.033451 -0.039866 -0.031587 -0.046156 -0.002758 -0.032316
PC5   0.143585  0.268222  0.309358  0.295992 -0.021423  0.162823 -0.030117
PC6  -0.067635 -0.225324 -0.119334 -0.128801 -0.080588 -0.270949  0.060366
PC7   0.126372  0.153634  0.323610  0.280628 -0.104650 -0.766114  0.062409
PC8  -0.083487 -0.369391 -0.111712 -0.102811 -0.059411  0.234186 -0.018771
PC9   0.070733  0.635989 -0.502211 -0.330848 -0.003236 -0.103672 -0.024518
PC10  0.039149  0.329636 -0.192834 -0.120983 -0.030850 -0.046499 -0.002360
PC11 -0.051640 -0.151760 -0.164001  0.019585  0.912500 -0.254798  0.015519
PC12  0.129459 -0.100793 -0.624348  0.751897 -0.123354  0.020193 -0.008683

           sk8       sk9      sk10      sk11      sk12      sk13
PC1   0.291481  0.234854  0.406509  0.282787  0.293839  0.429516
PC2   0.099394  0.086327  0.168557  0.104073  0.155922  0.236538
PC3   0.477371  0.079394 -0.373427  0.000937 -0.089114 -0.500784
PC4  -0.095896  0.105624  0.707258  0.138852  0.022035 -0.672875
PC5  -0.746663  0.111445 -0.196699  0.105288  0.259159 -0.096871
```

```
PC6   0.170746  0.160812 -0.285903  0.177867  0.786349 -0.208653
PC7   0.137519  0.252963 -0.034518  0.143796 -0.263787 -0.028619
PC8  -0.099978  0.714193 -0.170784  0.344654 -0.324104  0.041722
PC9  -0.067178 -0.042230 -0.102105  0.438945 -0.089894 -0.001708
PC10 -0.011806  0.550689  0.069338 -0.716713  0.112230  0.019667
PC11 -0.215919  0.034969  0.025239 -0.022616 -0.007803  0.012250
PC12  0.032981 -0.028696 -0.002688 -0.015985  0.006874 -0.000442
```

## 3.1 Importancia relativa de las variables sobre cada componente

Se puede apreciar segun el dataframe pca_vectors que : - Para PC1 las variables mas importantes en cuanto a peso relativo son : sk13 , sk10 ,sk7 - para PC2 las variables mas importante son : sk7, sk13,sk10,sk12,sk11

Podria concluirse que hay variables que tienen mayor peso sobre un componente que otras , indicando que ese grupo de variables podrian pertenecer a un factor como veremos mas adelante

Segun el criterio MLE el numero optimo de componentes es 12 , sin embargo si visualizamos el scree plot hay un momento en que la pendiente comienza a ser mas plana y no tan inclinada por lo que podria decirse segun el grafico que podria trabajarse con 7 componentes principales explicando un total acumulado de 0.835788847140444 de la varianza total

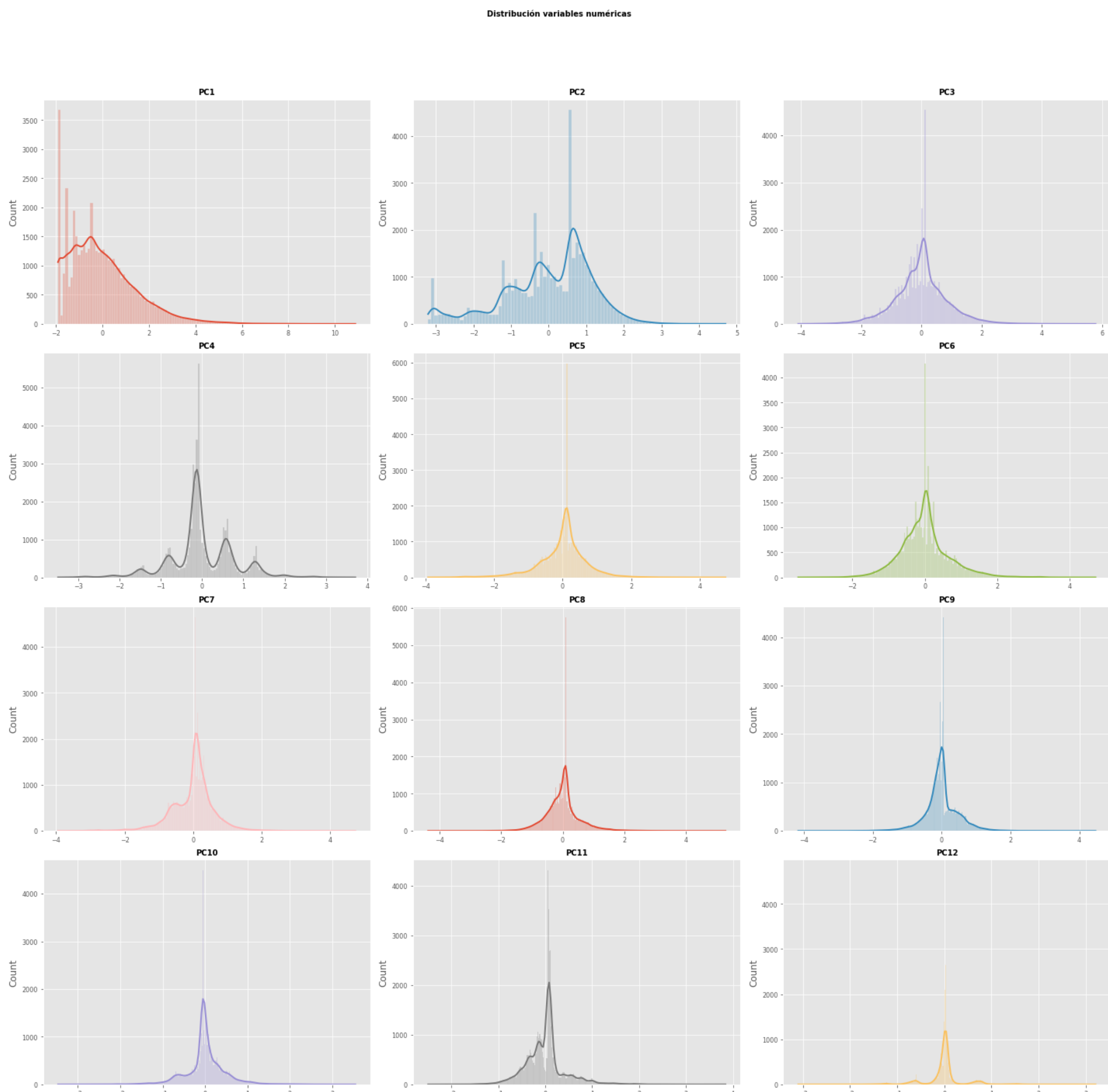
La descripción de cada componente se muestra a continuación.

[22]:
```python
pca_df = pd.DataFrame(data=pca_features,columns=['PC1', 'PC2',␣
↪'PC3',"PC4","PC5","PC6","PC7","PC8","PC9","PC10","PC11","PC12"])
pca_df.describe().apply(lambda s: s.apply('{0:.3f}'.format))
```

[22]:
```
             PC1        PC2        PC3        PC4        PC5        PC6  \
count  44502.000  44502.000  44502.000  44502.000  44502.000  44502.000
mean       0.000      0.000      0.000     -0.000      0.000      0.000
std        1.464      1.187      0.864      0.771      0.705      0.696
min       -1.910     -3.208     -4.086     -3.485     -3.926     -3.496
25%       -1.100     -0.714     -0.465     -0.255     -0.299     -0.400
50%       -0.260      0.185     -0.002     -0.097      0.094      0.018
75%        0.790      0.809      0.454      0.514      0.346      0.283
max       10.878      4.703      5.779      3.710      4.752      4.718

             PC7        PC8        PC9       PC10       PC11       PC12
count  44502.000  44502.000  44502.000  44502.000  44502.000  44502.000
mean       0.000     -0.000     -0.000     -0.000      0.000      0.000
std        0.622      0.573      0.492      0.470      0.433      0.397
min       -3.943     -4.363     -4.163     -3.469     -2.495     -3.101
25%       -0.311     -0.289     -0.212     -0.116     -0.221     -0.062
50%        0.065      0.026     -0.016     -0.038      0.051      0.021
75%        0.315      0.191      0.202      0.189      0.113      0.047
max        4.730      5.275      4.471      3.543      3.841      3.208
```

[23]:

```
# Gráfico de distribución para cada componente
# ==============================================================================
# Ajustar número de subplots en función del número de columnas
fig, axes = plt.subplots(nrows=4, ncols=3, figsize=(20, 20))
axes = axes.flat
columnas_numeric = pca_df.select_dtypes(include=['float64', 'int']).columns

for i, colum in enumerate(columnas_numeric):
        sns.histplot(
            data    = pca_df,
            x       = colum,
            stat    = "count",
            kde     = True,
            color   = (list(plt.rcParams['axes.prop_cycle'])*2)[i]["color"],
            line_kws= {'linewidth': 2},
            alpha   = 0.3,
            ax      = axes[i]
        )
        axes[i].set_title(colum, fontsize = 10, fontweight = "bold")
        axes[i].tick_params(labelsize = 8)
        axes[i].set_xlabel("")



fig.tight_layout()
plt.subplots_adjust(top = 0.9)
fig.suptitle('Distribución variables numéricas', fontsize = 10, fontweight =␣
 ↪"bold");
```




```
[24]: pca_df.corr().apply(lambda s: s.apply('{0:.3f}'.format))
```

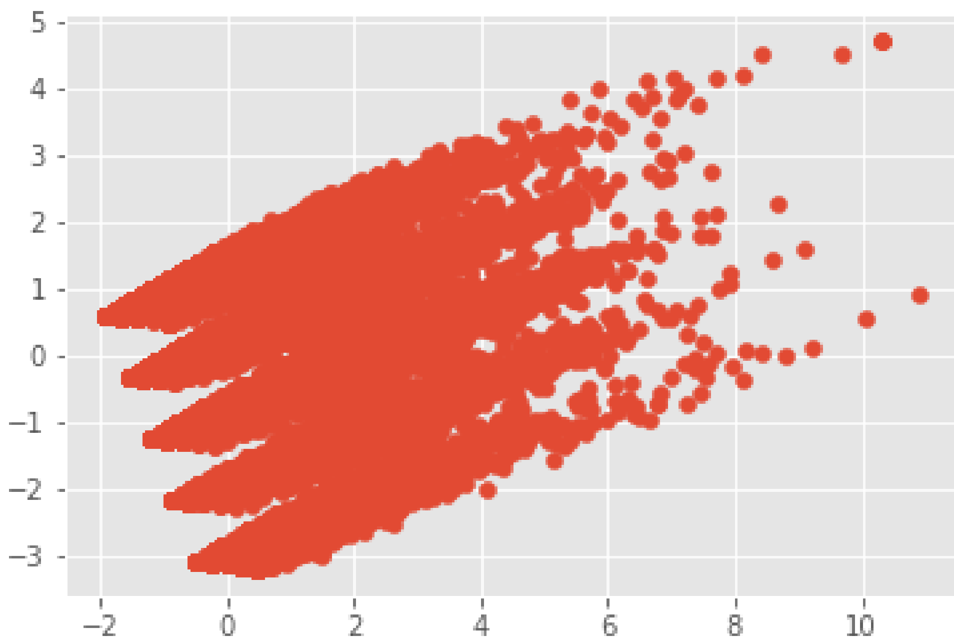
```
[24]:         PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9  \
PC1     1.000   0.000  -0.000  -0.000  -0.000   0.000   0.000   0.000   0.000
PC2     0.000   1.000   0.000   0.000  -0.000   0.000  -0.000   0.000   0.000
PC3    -0.000   0.000   1.000  -0.000  -0.000   0.000   0.000  -0.000   0.000
PC4    -0.000   0.000  -0.000   1.000   0.000   0.000   0.000   0.000  -0.000
PC5    -0.000  -0.000  -0.000   0.000   1.000  -0.000  -0.000   0.000   0.000
PC6     0.000   0.000   0.000   0.000  -0.000   1.000  -0.000   0.000  -0.000
PC7     0.000  -0.000   0.000   0.000  -0.000  -0.000   1.000   0.000   0.000
PC8     0.000   0.000  -0.000   0.000   0.000   0.000   0.000   1.000   0.000
PC9     0.000   0.000   0.000  -0.000   0.000  -0.000   0.000   0.000   1.000
PC10    0.000   0.000   0.000   0.000   0.000  -0.000   0.000   0.000  -0.000
```

```
PC11   0.000  -0.000   0.000   0.000   0.000  -0.000  -0.000   0.000  -0.000
PC12   0.000   0.000   0.000  -0.000   0.000  -0.000   0.000   0.000  -0.000

        PC10    PC11    PC12
PC1    0.000   0.000   0.000
PC2    0.000  -0.000   0.000
PC3    0.000   0.000   0.000
PC4    0.000   0.000  -0.000
PC5    0.000   0.000   0.000
PC6   -0.000  -0.000  -0.000
PC7    0.000  -0.000   0.000
PC8    0.000   0.000   0.000
PC9   -0.000  -0.000  -0.000
PC10   1.000  -0.000  -0.000
PC11  -0.000   1.000   0.000
PC12  -0.000   0.000   1.000
```

todos los vectores son ortogonales, por ende, no hay correlación entre ellos

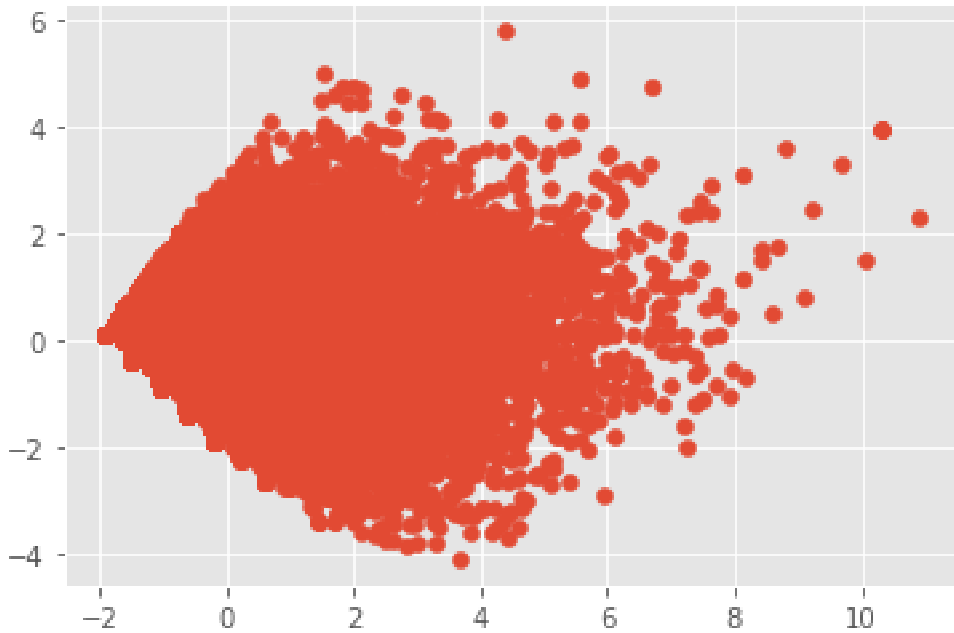
[25]:
```
plt.scatter(pca_df['PC1'],pca_df['PC2'])
```

[25]: <matplotlib.collections.PathCollection at 0x1beac3d4670>



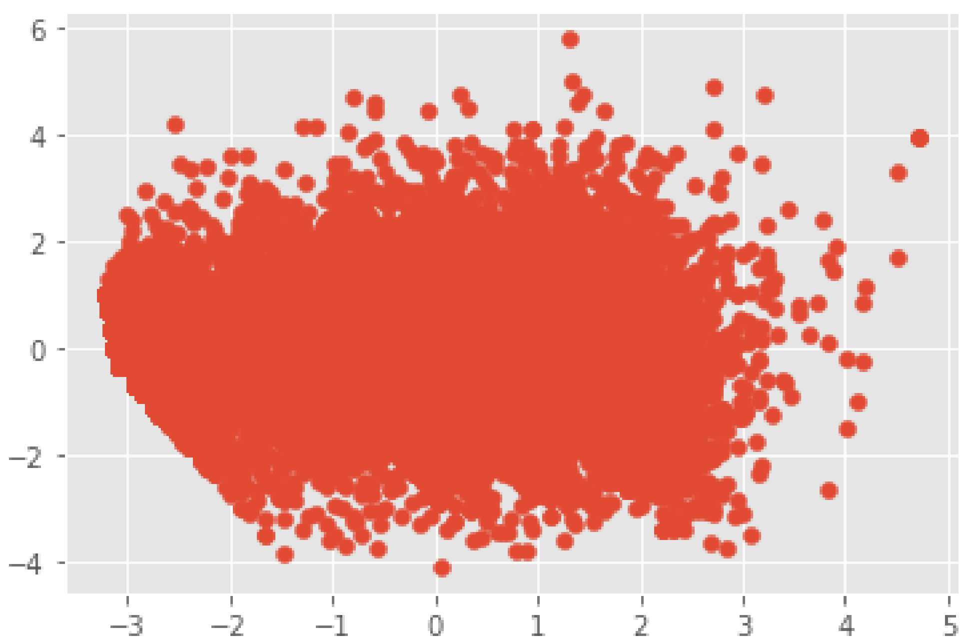

[26]:
```
plt.scatter(pca_df['PC1'],pca_df['PC3'])
```

[26]: <matplotlib.collections.PathCollection at 0x1beac5b11f0>



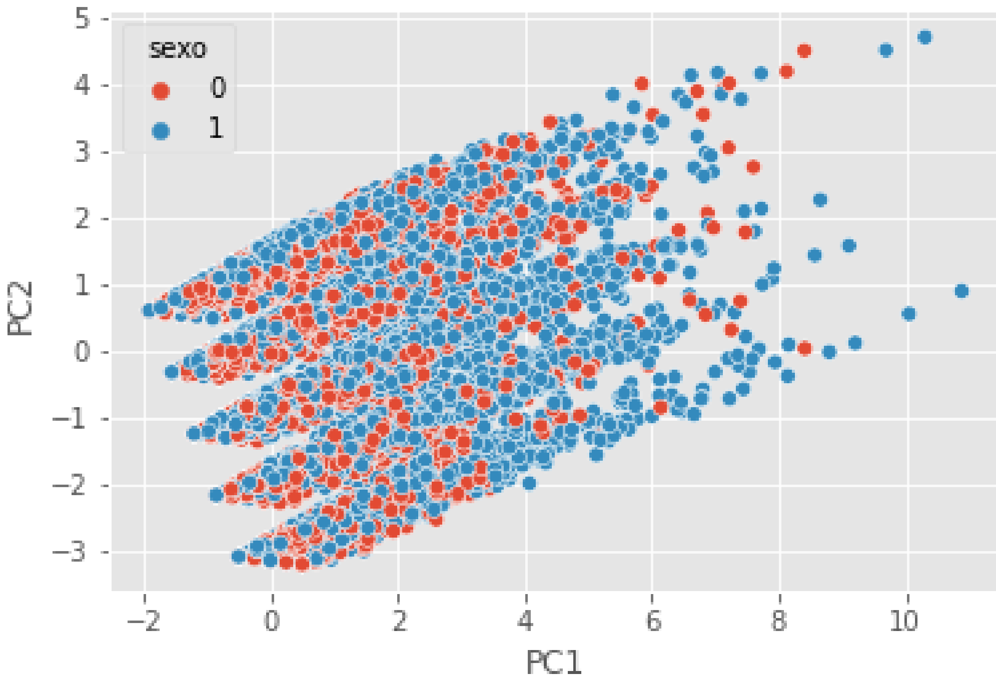

```
[27]: plt.scatter(pca_df['PC2'],pca_df['PC3'])
```

[27]: <matplotlib.collections.PathCollection at 0x1bead106790>



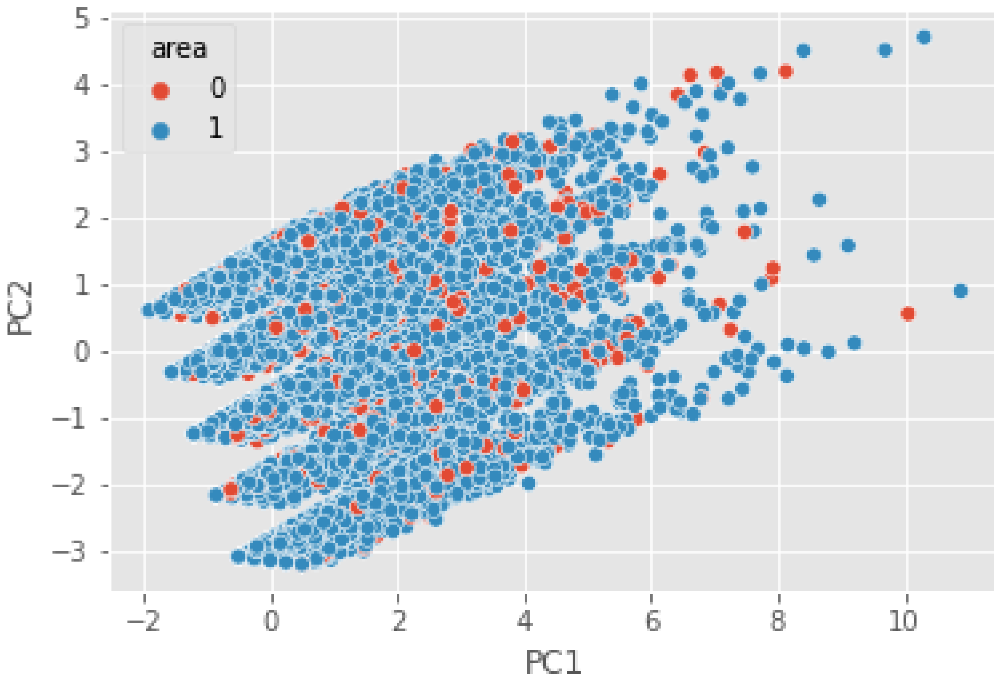
[28]:
```
pca_df['sexo'] = 0
pca_df['sexo'] = np.where(junaeb2['sexo'] > 0, 1, pca_df['sexo'])

sns.scatterplot('PC1', 'PC2', data=pca_df, hue='sexo')
```

[28]: <matplotlib.axes._subplots.AxesSubplot at 0x1beac336fa0>



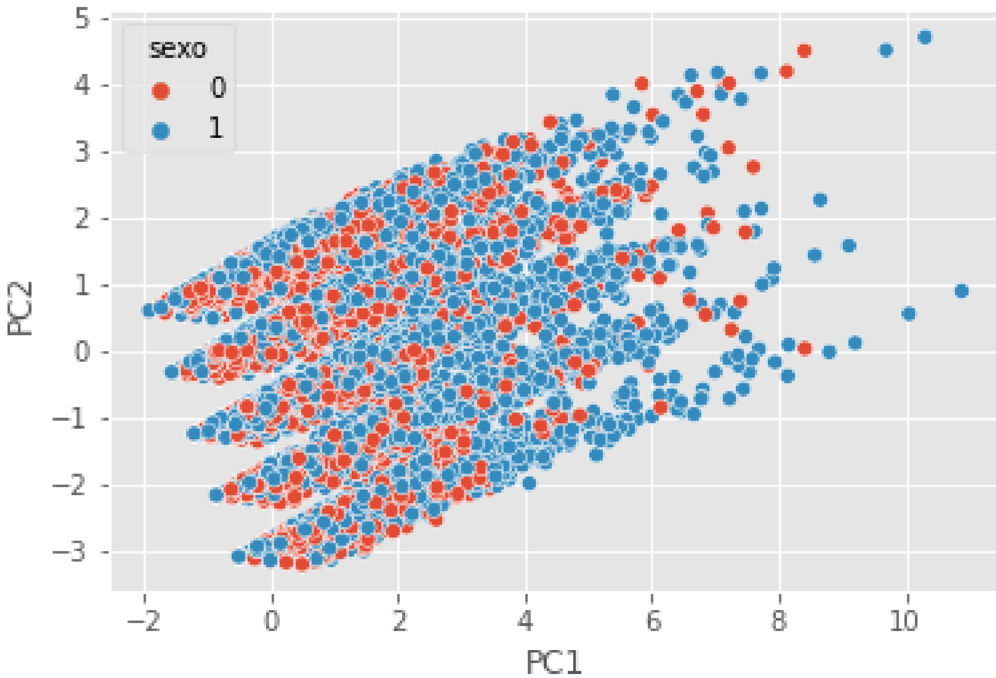

[29]:
```
pca_df['area'] = 0
pca_df['area'] = np.where(junaeb2['area'] > 0, 1, pca_df['area'])

sns.scatterplot('PC1', 'PC2', data=pca_df, hue='area')
```

[29]: <matplotlib.axes._subplots.AxesSubplot at 0x1bea6483cd0>




[30]:
```
pca_df['sexo'] = 0
pca_df['sexo'] = np.where(junaeb2['sexo'] > 0, 1, pca_df['sexo'])

sns.scatterplot('PC1', 'PC2', data=pca_df, hue='sexo')
```

[30]: <matplotlib.axes._subplots.AxesSubplot at 0x1beae2c4eb0>

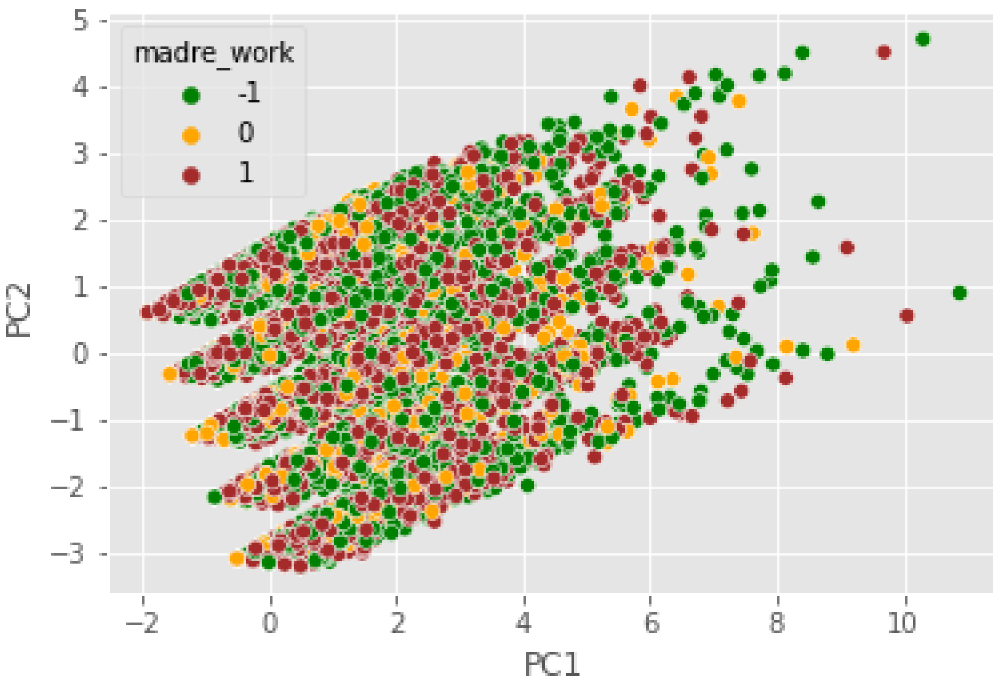



[31]:
```
pca_df['vive_madre'] = 0
pca_df['vive_madre'] = np.where(junaeb2['vive_madre'] > 0, 1,␣
 ↪pca_df['vive_madre'])

sns.scatterplot('PC1', 'PC2', data=pca_df, hue='vive_madre')
```

[31]: <matplotlib.axes._subplots.AxesSubplot at 0x1beac927a30>

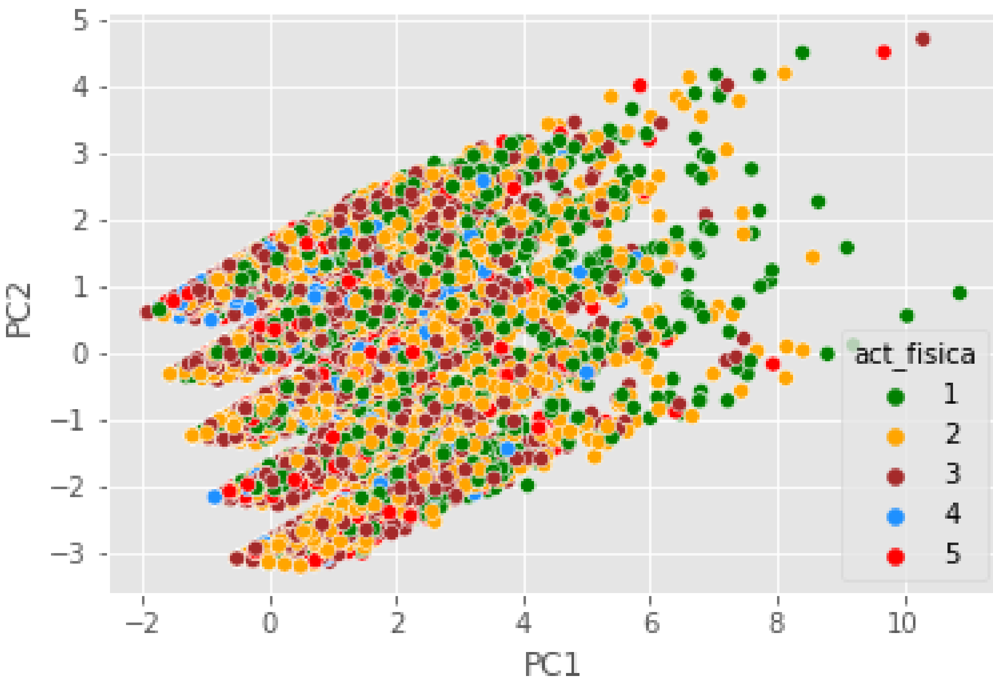

[32]:
```
pca_df['madre_work'] = 0
pca_df['madre_work'] = np.where(junaeb2['madre_work'] == 1, 1,␣
 ↪pca_df['madre_work'])
pca_df['madre_work'] = np.where(junaeb2['madre_work'] == -1, -1,␣
 ↪pca_df['madre_work'])

sns.scatterplot('PC1', 'PC2', data=pca_df,␣
 ↪hue='madre_work',palette=['green','orange','brown'])
```

[32]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1bea6fe8c10>
```




[33]:
```
pca_df['act_fisica'] = 0
pca_df['act_fisica'] = np.where(junaeb2['act_fisica'] == 1, 1,␣
 ↪pca_df['act_fisica'])
pca_df['act_fisica'] = np.where(junaeb2['act_fisica'] == 2, 2,␣
 ↪pca_df['act_fisica'])
pca_df['act_fisica'] = np.where(junaeb2['act_fisica'] == 3, 3,␣
 ↪pca_df['act_fisica'])
pca_df['act_fisica'] = np.where(junaeb2['act_fisica'] == 4, 4,␣
 ↪pca_df['act_fisica'])
pca_df['act_fisica'] = np.where(junaeb2['act_fisica'] == 5, 5,␣
 ↪pca_df['act_fisica'])

sns.scatterplot('PC1', 'PC2', data=pca_df, hue='act_fisica',␣
 ↪palette=['green','orange','brown','dodgerblue','red'], legend='full')
```

[33]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1beace63fa0>
```




se puede observar que no existen diferencias significativas entre grupos ya que no se observa claramente una separación entre ellos con respecto a ambos ejes.

[34]:
```python
## Si normalizamos los datos
def mean_norm(df_input):
    return df_input.apply(lambda x: (x-x.mean())/ x.std(), axis=0)

df_mean_norm = mean_norm(var_sk)
df_mean_norm.describe()
```

[34]:
```
                sk1           sk2           sk3           sk4           sk5  \
count  4.450200e+04  4.450200e+04  4.450200e+04  4.450200e+04  4.450200e+04
mean  -2.203382e-16 -1.130431e-16  2.746244e-17  1.967077e-16  9.005126e-17
std    1.000000e+00  1.000000e+00  1.000000e+00  1.000000e+00  1.000000e+00
min   -2.802821e-01 -5.909847e-01 -4.418237e-01 -4.363679e-01 -4.731034e-01
25%   -2.802821e-01 -5.909847e-01 -4.418237e-01 -4.363679e-01 -4.731034e-01
50%   -2.802821e-01 -5.909847e-01 -4.418237e-01 -4.363679e-01 -4.731034e-01
75%   -2.802821e-01  9.636645e-01 -4.418237e-01 -4.363679e-01 -4.731034e-01
max    1.050494e+01  5.627612e+00  6.566567e+00  6.742650e+00  6.679015e+00

                sk6           sk7           sk8           sk9          sk10  \
count  4.450200e+04  4.450200e+04  4.450200e+04  4.450200e+04  4.450200e+04
mean  -1.545561e-16  1.181524e-16  9.611854e-17 -3.257173e-17  1.085724e-16
std    1.000000e+00  1.000000e+00  1.000000e+00  1.000000e+00  1.000000e+00
min   -6.571768e-01 -1.017012e+00 -6.635656e-01 -4.929530e-01 -9.089731e-01
```

```
25%   -6.571768e-01 -1.017012e+00 -6.635656e-01 -4.929530e-01 -9.089731e-01
50%   -6.571768e-01 -1.950464e-01 -6.635656e-01 -4.929530e-01  1.645135e-01
75%    7.150868e-01  6.269191e-01  5.462339e-01 -4.929530e-01  1.645135e-01
max    4.831877e+00  2.270850e+00  4.175633e+00  5.690950e+00  3.384973e+00

               sk11          sk12          sk13
count  4.450200e+04  4.450200e+04  4.450200e+04
mean   1.251776e-16  7.791669e-17 -3.704236e-17
std    1.000000e+00  1.000000e+00  1.000000e+00
min   -5.701283e-01 -6.221762e-01 -7.001705e-01
25%   -5.701283e-01 -6.221762e-01 -7.001705e-01
50%   -5.701283e-01 -6.221762e-01 -7.001705e-01
75%    9.619882e-01  6.431599e-01  3.220724e-01
max    5.558338e+00  4.439168e+00  3.388801e+00
```

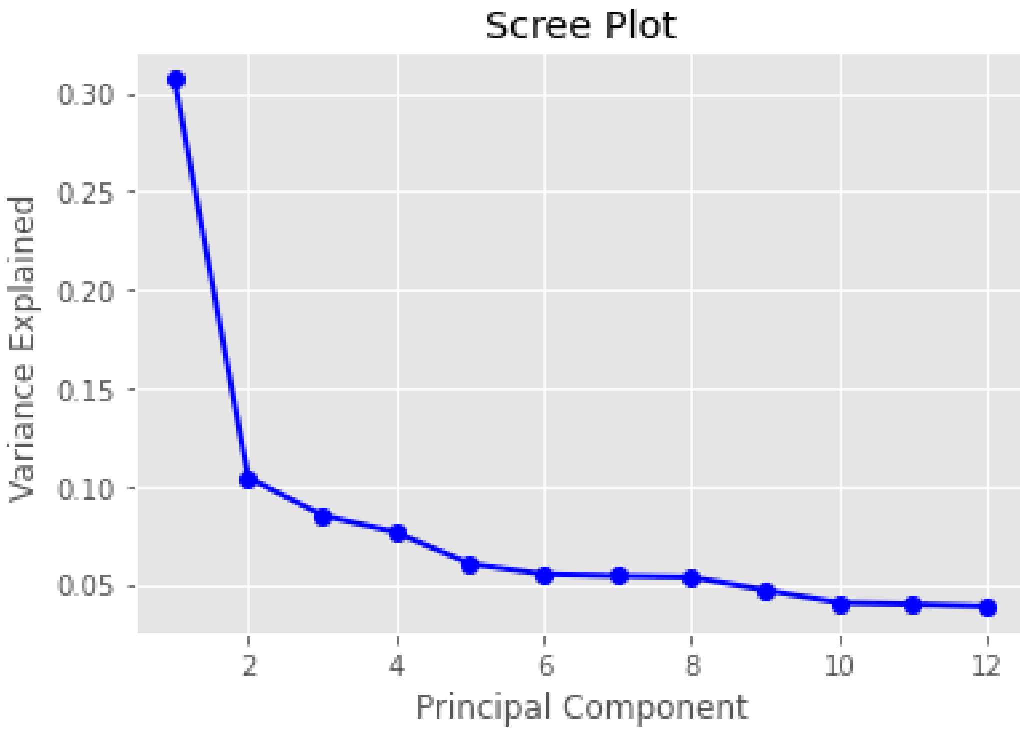
[35]:

```
pca = PCA(n_components=12)
pca_features = pca.fit_transform(df_mean_norm)
print(pca.explained_variance_ratio_)
```

```
[0.30723333 0.10439201 0.08525126 0.07644979 0.06046924 0.05518055
 0.05428047 0.05360694 0.0470409  0.04059575 0.03990748 0.03872587]
```

[36]:

```
#scree plot using explained variance proportion

PC_values = np.arange(pca.n_components_) + 1
plt.plot(PC_values, pca.explained_variance_ratio_, 'o-', linewidth=2,␣
 ↪color='blue')
plt.title('Scree Plot')
plt.xlabel('Principal Component')
plt.ylabel('Variance Explained')
plt.show()
```




[37]:
```
##Criterio MLE
pca = PCA(n_components='mle')
pca_features = pca.fit_transform(df_mean_norm)
print(pca.explained_variance_ratio_)
```

```
[0.30723333 0.10439201 0.08525126 0.07644979 0.06046924 0.05518055
 0.05428047 0.05360694 0.0470409  0.04059575 0.03990748 0.03872587]
```

[38]:
```
pca_vectors = pd.DataFrame(data = pca.components_,
                          columns=df_mean_norm.columns,
                          ␣
 ↪index=["PC1","PC2","PC3","PC4","PC5","PC6","PC7","PC8","PC9","PC10","PC11","PC12"])
pca_vectors
```

[38]:
```
            sk1       sk2       sk3       sk4       sk5       sk6       sk7  \
PC1   0.276396  0.320327  0.290478  0.318288  0.298207  0.266387  0.102880
PC2   0.303647  0.304357  0.266466  0.293335  0.210307  0.169534  0.104508
PC3   0.278695  0.040759  0.361099  0.311443 -0.370486 -0.445549 -0.431793
PC4   0.242956  0.049335 -0.059993 -0.011397 -0.344622 -0.001792  0.785706
PC5  -0.125028 -0.280401  0.146920  0.099917 -0.048960 -0.092096  0.153541
PC6   0.467324  0.240568 -0.350315 -0.190897 -0.045399 -0.520090  0.010475
PC7   0.123061  0.155605 -0.238702 -0.125883 -0.016727  0.405329 -0.348906
PC8   0.268908  0.192465 -0.288504 -0.164731  0.021957  0.153719 -0.118441
```

```
PC9   0.193620  0.006029 -0.230737 -0.103294 -0.072817  0.186780 -0.078231
PC10 -0.470616  0.677968  0.162376 -0.275631 -0.022319 -0.179493  0.043626
PC11  0.016812 -0.186801 -0.265133  0.204495  0.670632 -0.329148  0.059867
PC12 -0.127923  0.237140 -0.026034 -0.054730  0.287632 -0.242952  0.060446

           sk8       sk9      sk10      sk11      sk12      sk13
PC1   0.261728  0.290988  0.263789  0.328251  0.258510  0.259810
PC2  -0.025635 -0.232208 -0.391483 -0.321866 -0.380438 -0.343594
PC3  -0.389571  0.061267  0.022102  0.053263  0.069763  0.085723
PC4  -0.405200 -0.009870  0.168797  0.059478 -0.022725  0.043338
PC5   0.073324  0.654697 -0.230635  0.236136 -0.046923 -0.542736
PC6   0.472016  0.056328  0.085247  0.080103 -0.194923 -0.119411
PC7  -0.335932  0.182668  0.452092  0.144906 -0.386947 -0.283160
PC8  -0.213515 -0.062996 -0.335874  0.094216  0.688150 -0.311993
PC9  -0.103619  0.463023 -0.507390 -0.109843 -0.222359  0.555136
PC10 -0.134987  0.016024 -0.196818  0.329875 -0.121127  0.034897
PC11 -0.394590 -0.089445 -0.065824  0.317603 -0.121230  0.086235
PC12 -0.184415  0.407025  0.264036 -0.683649  0.200128 -0.038666
```

[39]:

```python
pca_df = pd.DataFrame(data=pca_features,columns=['PC1', 'PC2',
 'PC3',"PC4","PC5","PC6","PC7","PC8","PC9","PC10","PC11","PC12"])
pca_df.describe().apply(lambda s: s.apply('{0:.3f}'.format))
```

[39]:

```
             PC1        PC2        PC3        PC4        PC5        PC6  \
count  44502.000  44502.000  44502.000  44502.000  44502.000  44502.000
mean      -0.000     -0.000      0.000     -0.000     -0.000      0.000
std        1.999      1.165      1.053      0.997      0.887      0.847
min       -2.042     -8.224     -6.875     -5.240     -5.077     -5.961
25%       -1.471     -0.592     -0.595     -0.630     -0.451     -0.415
50%       -0.535      0.141      0.126     -0.080      0.100      0.058
75%        0.858      0.553      0.584      0.659      0.385      0.349
max       19.428      9.805      9.674      5.408      6.525      7.434

             PC7        PC8        PC9       PC10       PC11       PC12
count  44502.000  44502.000  44502.000  44502.000  44502.000  44502.000
mean       0.000      0.000      0.000     -0.000      0.000      0.000
std        0.840      0.835      0.782      0.726      0.720      0.710
min       -4.743     -4.818     -5.190     -6.107     -5.358     -4.833
25%       -0.485     -0.471     -0.435     -0.377     -0.368     -0.306
50%        0.032      0.039      0.066     -0.013      0.101     -0.034
75%        0.475      0.372      0.358      0.283      0.309      0.374
max        5.236      5.599      5.430      6.454      6.633      5.064
```

# 4 EFA

[40]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import statsmodels.formula.api as smf
import sklearn
import scipy
from scipy.linalg import eigh, cholesky
from scipy.stats import norm
import linearmodels.panel as lmp
from pylab import plot, show, axis, subplot, xlabel, ylabel, grid
import semopy
import seaborn as sns
from factor_analyzer import FactorAnalyzer
from sklearn.decomposition import PCA

%matplotlib inline
```

Ser egoista modificamos sus valores ya que Si una carateristica es negativa debe ser invertida en la escala, de tal forma que todas las variables representen aspectos positivos.

[41]:
```
## Ser egoista modificamos sus valores
var_sk["sk7"].replace(4,2,inplace=True)
var_sk["sk7"].replace(5,1,inplace=True)
var_sk["sk7"].replace(2,4,inplace=True)
var_sk["sk7"].replace(1,5,inplace=True)
```

[42]:
```
fa = FactorAnalyzer(rotation='promax')
fa.fit(var_sk)
```

[42]:
```
FactorAnalyzer(rotation_kwargs={})
```

[43]:
```
#Indica que factores pesan y en que dirección
fa.loadings_



efa_vectors = pd.DataFrame(data = fa.loadings_,
 ␣
 ↪index=["sk1","sk2","sk3","sk4","sk5","sk6","sk7","sk8","sk9","sk10","sk11","sk12","sk13"])
efa_vectors
```

[43]:
```
            0         1         2
sk1   0.016223  0.606457 -0.041318
sk2  -0.022246  0.493310  0.227285
sk3   0.025796  0.637033 -0.038952
```

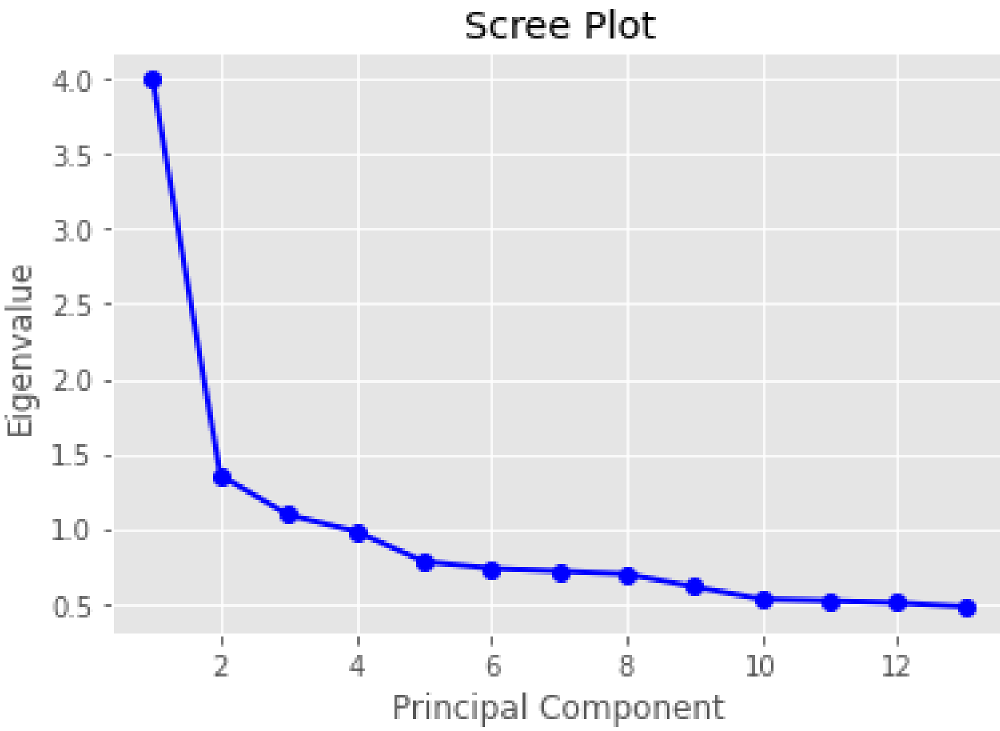
```
sk4  -0.004842  0.738683 -0.025207
sk5  -0.176251  0.002683  0.884808
sk6   0.015135  0.072977  0.474201
sk7  -0.130063 -0.002210 -0.080738
sk8   0.147399 -0.087711  0.499505
sk9   0.486060  0.097721  0.017551
sk10  0.641346 -0.043450 -0.049517
sk11  0.709336  0.034740 -0.029819
sk12  0.577990 -0.028291 -0.018118
sk13  0.537767  0.021121 -0.028952
```

El numero optimo de factores son 3

[44]:
```
fa.get_eigenvalues()
```

[44]:
```
(array([3.99962771, 1.35620325, 1.09229343, 0.98183971, 0.78110224,
        0.73386953, 0.71649271, 0.69706082, 0.61100148, 0.5286527 ,
        0.51829849, 0.50466929, 0.47888864]),
 array([ 3.41023023,  0.78599957,  0.60789792,  0.17836043,  0.09858874,
         0.06858721,  0.03691868,  0.00593339, -0.02781041, -0.05358291,
        -0.07616187, -0.09948733, -0.17644849]))
```

[45]:
```
values = np.arange(1,14)
eigenvalues = pd.DataFrame(data=fa.get_eigenvalues())
plt.plot(values, eigenvalues.loc[0], 'o-', linewidth=2, color='blue')
plt.title('Scree Plot')
plt.xlabel('Principal Component')
plt.ylabel('Eigenvalue')
plt.show()
```




[46]:
```
#matriz de varianza-covarianza
#3 elementos:
#-varianza de forma cruda
#-proporcion explicada de cada factor
#-proporción acumulada

fa.get_factor_variance()
```

[46]:
```
(array([1.8453857 , 1.5895295 , 1.32410995]),
 array([0.14195275, 0.1222715 , 0.10185461]),
 array([0.14195275, 0.26422425, 0.36607886]))
```

[47]:
```
print(semopy.efa.explore_cfa_model(var_sk, pval=0.05))
```

```
eta1 =~ sk11 + sk9 + sk10 + sk12
eta2 =~ sk4 + sk2 + sk5 + sk3 + sk1 + sk6 + sk8
eta3 =~ sk2 + sk6 + sk10 + sk7
```

[48]:
```
efa_vectors
```

[48]:

| | 0 | 1 | 2 |
|---|---|---|---|
| sk1 | 0.016223 | 0.606457 | -0.041318 |

```
sk2  -0.022246  0.493310  0.227285
sk3   0.025796  0.637033 -0.038952
sk4  -0.004842  0.738683 -0.025207
sk5  -0.176251  0.002683  0.884808
sk6   0.015135  0.072977  0.474201
sk7  -0.130063 -0.002210 -0.080738
sk8   0.147399 -0.087711  0.499505
sk9   0.486060  0.097721  0.017551
sk10  0.641346 -0.043450 -0.049517
sk11  0.709336  0.034740 -0.029819
sk12  0.577990 -0.028291 -0.018118
sk13  0.537767  0.021121 -0.028952
```

Se utilizara la matriz de pesos de cada factor donde cada variable pertenece a un solo factor y se decide a que factor mediante la variable que tenga mas peso sobre el factor, asi se tiene siguiente modelo :

eta1 =~ sk11 + sk9 + sk10 + sk7 + sk12 + sk13

eta2 =~ sk4 + sk2 + sk3 + sk1 + sk6

eta3 =~ sk6 + sk5 + sk8

## 5 General CFA

[49]:
```
Xf = var_sk

mod = """
# measurement model
eta1 =~ sk11 + sk9 + sk10 + sk7 + sk12 + sk13
eta2 =~ sk4 + sk2 + sk3 + sk1
eta3 =~ sk5 + sk8 + sk6
    """

model = semopy.Model(mod) #se entrega el modelo
out=model.fit(Xf)
print(out)

#output
#tipo de función utilizada
#algoritmo de optimización
#valor final de la función
#numero de iteraciones
#parametros igual a los pesos relativos
#para cada factor hay un parametro estimado
```

```
Name of objective: MLW
Optimization method: SLSQP
```

```
Optimization successful.
Optimization terminated successfully
Objective value: 0.156
Number of iterations: 35
Params: 0.796 1.120 -0.305 0.914 1.081 1.078 0.914 0.561 1.136 1.010 0.699 0.151
0.200 0.475 0.160 0.592 0.440 0.090 0.239 0.590 0.279 0.205 0.366 0.221 0.108
0.110 0.162 0.150 0.102
```

[50]:

```python
model.inspect(mode='list', what="names", std_est=True)
```

[50]:

| | lval | op | rval | Estimate | Est. Std | Std. Err | z-value | p-value |
|---|---|---|---|---|---|---|---|---|
| 0 | sk11 | ~ | eta1 | 1.000000 | 0.719789 | - | - | - |
| 1 | sk9 | ~ | eta1 | 0.796146 | 0.578252 | 0.007979 | 99.777867 | 0.0 |
| 2 | sk10 | ~ | eta1 | 1.120498 | 0.565192 | 0.011443 | 97.92038 | 0.0 |
| 3 | sk7 | ~ | eta1 | -0.305009 | -0.183069 | 0.00905 | -33.703474 | 0.0 |
| 4 | sk12 | ~ | eta1 | 0.914277 | 0.543503 | 0.00965 | 94.748434 | 0.0 |
| 5 | sk13 | ~ | eta1 | 1.080755 | 0.518964 | 0.01187 | 91.04675 | 0.0 |
| 6 | sk4 | ~ | eta2 | 1.000000 | 0.695728 | - | - | - |
| 7 | sk2 | ~ | eta2 | 1.078383 | 0.649815 | 0.009951 | 108.371458 | 0.0 |
| 8 | sk3 | ~ | eta2 | 0.913827 | 0.620692 | 0.008717 | 104.82871 | 0.0 |
| 9 | sk1 | ~ | eta2 | 0.561322 | 0.586646 | 0.005595 | 100.323096 | 0.0 |
| 10 | sk5 | ~ | eta3 | 1.000000 | 0.718571 | - | - | - |
| 11 | sk8 | ~ | eta3 | 1.136319 | 0.552490 | 0.013134 | 86.514464 | 0.0 |
| 12 | sk6 | ~ | eta3 | 1.009705 | 0.556913 | 0.01161 | 86.965304 | 0.0 |
| 13 | eta1 | ~~ | eta1 | 0.220737 | 1.000000 | 0.002926 | 75.452184 | 0.0 |
| 14 | eta1 | ~~ | eta2 | 0.108261 | 0.594420 | 0.001472 | 73.535278 | 0.0 |
| 15 | eta1 | ~~ | eta3 | 0.109743 | 0.581168 | 0.001546 | 70.989102 | 0.0 |
| 16 | eta3 | ~~ | eta3 | 0.161537 | 1.000000 | 0.002357 | 68.546914 | 0.0 |
| 17 | eta2 | ~~ | eta2 | 0.150273 | 1.000000 | 0.002062 | 72.866704 | 0.0 |
| 18 | eta2 | ~~ | eta3 | 0.102088 | 0.655234 | 0.001346 | 75.834931 | 0.0 |
| 19 | sk13 | ~~ | sk13 | 0.699486 | 0.730676 | 0.005277 | 132.563385 | 0.0 |
| 20 | sk5 | ~~ | sk5 | 0.151311 | 0.483656 | 0.001792 | 84.416885 | 0.0 |
| 21 | sk3 | ~~ | sk3 | 0.200239 | 0.614742 | 0.00165 | 121.375105 | 0.0 |
| 22 | sk8 | ~~ | sk8 | 0.474739 | 0.694754 | 0.003867 | 122.757553 | 0.0 |
| 23 | sk4 | ~~ | sk4 | 0.160184 | 0.515962 | 0.001493 | 107.321549 | 0.0 |
| 24 | sk7 | ~~ | sk7 | 0.592198 | 0.966486 | 0.004011 | 147.629895 | 0.0 |
| 25 | sk12 | ~~ | sk12 | 0.440119 | 0.704604 | 0.003379 | 130.235165 | 0.0 |
| 26 | sk1 | ~~ | sk1 | 0.090231 | 0.655847 | 0.000716 | 125.994347 | 0.0 |
| 27 | sk2 | ~~ | sk2 | 0.239100 | 0.577740 | 0.00205 | 116.636226 | 0.0 |
| 28 | sk10 | ~~ | sk10 | 0.590431 | 0.680558 | 0.004616 | 127.918594 | 0.0 |
| 29 | sk9 | ~~ | sk9 | 0.278520 | 0.665625 | 0.002204 | 126.390022 | 0.0 |
| 30 | sk11 | ~~ | sk11 | 0.205317 | 0.481904 | 0.002047 | 100.306409 | 0.0 |
| 31 | sk6 | ~~ | sk6 | 0.366303 | 0.689848 | 0.003 | 122.105112 | 0.0 |

[51]:

```python
semopy.calc_stats(model)
```

[51]:
```
         DoF  DoF Baseline         chi2  chi2 p-value  chi2 Baseline       CFI  \
Value   62            78  6920.382288           0.0   120036.27426  0.942827

            GFI     AGFI       NFI       TLI     RMSEA        AIC         BIC  \
Value  0.942348  0.92747  0.942348  0.928073  0.049857  57.688986  310.084378

         LogLik
Value  0.155507
```

eta3 tenga como nombre = sociable - sk5: juega con otros (1: siempre - 5: nunca) - sk6: comparte sus cosas con otros (1: siempre - 5: nunca
- sk8: participa en juegos grupales (1: siempre - 5: nunca)

eta1 tenga como nombre = creatividad

- sk9: hace preguntas a adultos (1: siempre - 5: nunca)
- sk10: tiene interes por libros (1: siempre - 5: nunca)
- sk11: tiene interes por su entorno (1: siempre - 5: nunca)
- sk12: juega a armar y desarmar cosas (1: siempre - 5: nunca)
- sk13: tiene expresiones artisticas (1: siempre - 5: nunca)
- sk7: es agresivo (1: siempre - 5: nunca)

eta2 tenga como nombre = inteligencia emocional

- sk1: muestra afecto a padres (1: siempre - 5: nunca)
- sk2: muestra afecto a sus pares (1: siempre - 5: nunca)
- sk3: expresa sus sentimientos (1: siempre - 5: nunca)
- sk4: usa gestos para mostrar sentimientos (1: siempre - 5: nunca)

## 6 Complete sem

[52]:
```python
# incluyendo imce, act_fisica y area
var_sk["sexo"] = junaeb2["sexo"].copy()
var_sk["imce"] = junaeb2["imce"].copy()
var_sk["act_fisica"] = junaeb2["act_fisica"].copy()
var_sk["area"] = junaeb2["area"].copy()
```

[53]:
```python
mod = """
# measurement model
act_fisica =~ sexo + imce + area
creatividad =~ sk11 + sk9 + sk10 + sk7 + sk12 + sk13
inteligencia_emocional =~ sk4 + sk2 + sk3 + sk1
sociable =~ sk5 + sk8 + sk6
#regression
act_fisica ~ creatividad + inteligencia_emocional + sociable
    """

model = semopy.Model(mod) #se entrega el modelo
```

```python
out=model.fit(Xf)
print(out)
```

```
Name of objective: MLW
Optimization method: SLSQP
Optimization successful.
Optimization terminated successfully
Objective value: 0.259
Number of iterations: 47
Params: 0.414 -0.016 0.798 1.152 -0.319 0.915 1.130 1.080 0.914 0.562 1.137
1.011 0.216 0.013 -0.077 1.062 0.105 0.683 0.151 0.200 0.136 0.475 0.160 0.591
0.444 0.090 0.239 0.582 0.281 0.087 0.211 0.366 0.215 0.108 0.107 0.161 0.102
0.150
```

[54]:
```python
model.inspect(mode='list', what="names", std_est=True)
```

[54]:

| | lval | op | rval | Estimate | Est. Std | \ |
|---|---|---|---|---|---|---|
| 0 | act_fisica | ~ | creatividad | 0.215855 | 0.298152 | |
| 1 | act_fisica | ~ | inteligencia_emocional | 0.012928 | 0.014922 | |
| 2 | act_fisica | ~ | sociable | -0.076654 | -0.091760 | |
| 3 | sexo | ~ | act_fisica | 1.000000 | 0.672973 | |
| 4 | imce | ~ | act_fisica | 0.414351 | 0.133732 | |
| 5 | area | ~ | act_fisica | -0.016124 | -0.018323 | |
| 6 | sk11 | ~ | creatividad | 1.000000 | 0.710202 | |
| 7 | sk9 | ~ | creatividad | 0.798190 | 0.572032 | |
| 8 | sk10 | ~ | creatividad | 1.152362 | 0.573503 | |
| 9 | sk7 | ~ | creatividad | -0.318844 | -0.188825 | |
| 10 | sk12 | ~ | creatividad | 0.915446 | 0.536963 | |
| 11 | sk13 | ~ | creatividad | 1.129886 | 0.535422 | |
| 12 | sk4 | ~ | inteligencia_emocional | 1.000000 | 0.695280 | |
| 13 | sk2 | ~ | inteligencia_emocional | 1.080068 | 0.650415 | |
| 14 | sk3 | ~ | inteligencia_emocional | 0.913966 | 0.620286 | |
| 15 | sk1 | ~ | inteligencia_emocional | 0.561661 | 0.586631 | |
| 16 | sk5 | ~ | sociable | 1.000000 | 0.718309 | |
| 17 | sk8 | ~ | sociable | 1.136690 | 0.552516 | |
| 18 | sk6 | ~ | sociable | 1.010507 | 0.557091 | |
| 19 | act_fisica | ~~ | act_fisica | 0.104838 | 0.930803 | |
| 20 | creatividad | ~~ | creatividad | 0.214887 | 1.000000 | |
| 21 | creatividad | ~~ | sociable | 0.108371 | 0.581918 | |
| 22 | creatividad | ~~ | inteligencia_emocional | 0.106864 | 0.595108 | |
| 23 | sociable | ~~ | sociable | 0.161395 | 1.000000 | |
| 24 | sociable | ~~ | inteligencia_emocional | 0.101976 | 0.655273 | |
| 25 | inteligencia_emocional | ~~ | inteligencia_emocional | 0.150058 | 1.000000 | |
| 26 | imce | ~~ | imce | 1.061915 | 0.982116 | |
| 27 | sk13 | ~~ | sk13 | 0.682612 | 0.713324 | |
| 28 | sk5 | ~~ | sk5 | 0.151406 | 0.484032 | |
| 29 | sk3 | ~~ | sk3 | 0.200439 | 0.615246 | |

| | | | | | |
|---|---|---|---|---|---|
| 30 | sexo | ~~ | sexo | 0.136062 | 0.547107 |
| 31 | sk8 | ~~ | sk8 | 0.474569 | 0.694726 |
| 32 | sk4 | ~~ | sk4 | 0.160354 | 0.516585 |
| 33 | sk7 | ~~ | sk7 | 0.590852 | 0.964345 |
| 34 | sk12 | ~~ | sk12 | 0.444496 | 0.711671 |
| 35 | sk1 | ~~ | sk1 | 0.090217 | 0.655863 |
| 36 | sk2 | ~~ | sk2 | 0.238740 | 0.576960 |
| 37 | sk10 | ~~ | sk10 | 0.582239 | 0.671095 |
| 38 | sk9 | ~~ | sk9 | 0.281484 | 0.672779 |
| 39 | area | ~~ | area | 0.087189 | 0.999664 |
| 40 | sk11 | ~~ | sk11 | 0.211149 | 0.495613 |
| 41 | sk6 | ~~ | sk6 | 0.366223 | 0.689649 |

| | Std. Err | z-value | p-value |
|---|---|---|---|
| 0 | 0.008979 | 24.04014 | 0.0 |
| 1 | 0.012035 | 1.07416 | 0.282751 |
| 2 | 0.012093 | -6.338814 | 0.0 |
| 3 | - | - | - |
| 4 | 0.066084 | 6.270041 | 0.0 |
| 5 | 0.006519 | -2.473441 | 0.013382 |
| 6 | - | - | - |
| 7 | 0.008103 | 98.506268 | 0.0 |
| 8 | 0.011674 | 98.713169 | 0.0 |
| 9 | 0.009178 | -34.738435 | 0.0 |
| 10 | 0.009798 | 93.428297 | 0.0 |
| 11 | 0.012123 | 93.199457 | 0.0 |
| 12 | - | - | - |
| 13 | 0.009964 | 108.396535 | 0.0 |
| 14 | 0.008726 | 104.736679 | 0.0 |
| 15 | 0.005601 | 100.285204 | 0.0 |
| 16 | - | - | - |
| 17 | 0.013138 | 86.517377 | 0.0 |
| 18 | 0.011617 | 86.983745 | 0.0 |
| 19 | 0.018027 | 5.815747 | 0.0 |
| 20 | 0.002889 | 74.378521 | 0.0 |
| 21 | 0.001532 | 70.727024 | 0.0 |
| 22 | 0.001459 | 73.233621 | 0.0 |
| 23 | 0.002355 | 68.527557 | 0.0 |
| 24 | 0.001345 | 75.804619 | 0.0 |
| 25 | 0.002061 | 72.808802 | 0.0 |
| 26 | 0.007765 | 136.757303 | 0.0 |
| 27 | 0.005206 | 131.109838 | 0.0 |
| 28 | 0.001792 | 84.505589 | 0.0 |
| 29 | 0.001651 | 121.430448 | 0.0 |
| 30 | 0.018049 | 7.538313 | 0.0 |
| 31 | 0.003866 | 122.754706 | 0.0 |
| 32 | 0.001493 | 107.417819 | 0.0 |

```
33  0.004005  147.532236       0.0
34  0.003394  130.960976       0.0
35  0.000716  125.991648       0.0
36  0.002049  116.523986       0.0
37  0.004582  127.061083       0.0
38  0.002212  127.232871       0.0
39  0.000585  149.108408       0.0
40  0.002049  103.066465       0.0
41     0.003  122.079502       0.0
```

[55]:
```
semopy.calc_stats(model)
```

[55]:
```
       DoF  DoF Baseline          chi2  chi2 p-value  chi2 Baseline       CFI  \
Value   98           120  11525.011114           0.0  125976.114565  0.909206

            GFI      AGFI       NFI       TLI     RMSEA        AIC  \
Value  0.908514  0.887977  0.908514  0.888823  0.051188  75.482045

              BIC    LogLik
Value  406.207043  0.258977
```

Se puede apreciar que las latentes creatividad y sociable tienen valor p inferior a 5% por lo que son significativas para el modelo de regression que tiene como variable dependiente act_fisica. por otro lado inteligencia emocional no es significativa para el modelo de regresion.

La creatividad tiene el coeficiente más alto 0.215855, lo que significa que un cambio en la creatividad tendrá el mayor impacto en la actividad fisica. Un aumento en la variable latente creatividad tiene un impacto positivo en la actividad fisica.

La variable latente sociable tiene el coeficiente -0.076654 , lo que significa que un cambio en la variable latente sociable tendrá un impacto en la actividad fisica. es decir un aumento en la variable latente sociable tiene un impacto negativo en la actividad fisica.