

Tarea4_Cuevas_Reyes

December 19, 2022

Section 6: Structural Equation Modelling

Tarea 4

Instrucciones

Los resultados de los ejercicios propuestos se deben entregar como un notebook por correo electrónico a *juancaros@udec.cl* el día 6/12 hasta las 21:00. Utilizar la base de datos *junaeb2.csv*. La base corresponde a observaciones tomadas de estudiantes de colegio. Las variables tienen la siguiente descripción:

- sexo: sexo del estudiante
- edad: edad del estudiante (meses)
- imce: índice de masa corporal estandarizado
- vive_padre: si el padre vive en el hogar
- vive_madre: si la madre vive en el hogar
- area: urbana=1, rural=0
- 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)
- sk5: juega con otros (1: siempre - 5: nunca)
- sk6: comparte sus cosas con otros (1: siempre - 5: nunca)
- sk7: es agresivo (1: siempre - 5: nunca)
- sk8: participa en juegos grupales (1: siempre - 5: nunca)
- sk9: hace preguntas a adultos (1: siempre - 5: nunca)
- sk10: tiene interés por libros (1: siempre - 5: nunca)
- sk11: tiene interés por su entorno (1: siempre - 5: nunca)
- sk12: juega a armar y desarmar cosas (1: siempre - 5: nunca)
- sk13: tiene expresiones artísticas (1: siempre - 5: nunca)
- act_fisica: frecuencia actividad física (1: nunca - 5: 5 o más veces a la semana)
- educm: años de escolaridad de la madre
- educp: años de escolaridad del padre
- madre_work: si la madre trabaja (-1: labor doméstica, 0: desempleada, 1: empleada)

```
[ ]: !pip install linearmodels  
!pip install semopy  
!pip install factor_analyzer
```

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab>

```
wheels/public/simple/
Collecting linearmodels
    Downloading
linearmodels-4.27-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.5
MB)
|           | 1.5 MB 5.0 MB/s
Requirement already satisfied: statsmodels>=0.11 in
/usr/local/lib/python3.8/dist-packages (from linearmodels) (0.12.2)
Collecting pyhdfe>=0.1
    Downloading pyhdfe-0.1.1-py3-none-any.whl (18 kB)
Requirement already satisfied: Cython>=0.29.21 in /usr/local/lib/python3.8/dist-
packages (from linearmodels) (0.29.32)
Collecting property-cached>=1.6.3
    Downloading property_cached-1.6.4-py2.py3-none-any.whl (7.8 kB)
Collecting formulaic~0.3.2
    Downloading formulaic-0.3.4-py3-none-any.whl (68 kB)
|           | 68 kB 4.1 MB/s
Collecting setuptools-scm<7.0.0,>=6.4.2
    Downloading setuptools_scm-6.4.2-py3-none-any.whl (37 kB)
Collecting mypy-extensions>=0.4
    Downloading mypy_extensions-0.4.3-py2.py3-none-any.whl (4.5 kB)
Requirement already satisfied: pandas>=0.24 in /usr/local/lib/python3.8/dist-
packages (from linearmodels) (1.3.5)
Requirement already satisfied: scipy>=1.2 in /usr/local/lib/python3.8/dist-
packages (from linearmodels) (1.7.3)
Requirement already satisfied: numpy>=1.16 in /usr/local/lib/python3.8/dist-
packages (from linearmodels) (1.21.6)
Collecting interface-meta<2.0.0,>=1.2.0
    Downloading interface_meta-1.3.0-py3-none-any.whl (14 kB)
Requirement already satisfied: astor>=0.8 in /usr/local/lib/python3.8/dist-
packages (from formulaic~0.3.2->linearmodels) (0.8.1)
Requirement already satisfied: wrapt>=1.0 in /usr/local/lib/python3.8/dist-
packages (from formulaic~0.3.2->linearmodels) (1.14.1)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.8/dist-
packages (from pandas>=0.24->linearmodels) (2022.6)
Requirement already satisfied: python-dateutil>=2.7.3 in
/usr/local/lib/python3.8/dist-packages (from pandas>=0.24->linearmodels) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.8/dist-
packages (from python-dateutil>=2.7.3->pandas>=0.24->linearmodels) (1.15.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.8/dist-
packages (from setuptools-scm<7.0.0,>=6.4.2->linearmodels) (21.3)
Requirement already satisfied: tomli>=1.0.0 in /usr/local/lib/python3.8/dist-
packages (from setuptools-scm<7.0.0,>=6.4.2->linearmodels) (2.0.1)
Requirement already satisfied: setuptools in /usr/local/lib/python3.8/dist-
packages (from setuptools-scm<7.0.0,>=6.4.2->linearmodels) (57.4.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in
/usr/local/lib/python3.8/dist-packages (from packaging>=20.0->setuptools-
scm<7.0.0,>=6.4.2->linearmodels) (3.0.9)
```

```

Requirement already satisfied: patsy>=0.5 in /usr/local/lib/python3.8/dist-
packages (from statsmodels>=0.11->linearmodels) (0.5.3)
Installing collected packages: interface-meta, setuptools-scm, pyhdfe, property-
cached, mypy-extensions, formulaic, linearmodels
Successfully installed formulaic-0.3.4 interface-meta-1.3.0 linearmodels-4.27
mypy-extensions-0.4.3 property-cached-1.6.4 pyhdfe-0.1.1 setuptools-scm-6.4.2
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Collecting semopy
    Downloading semopy-2.3.9.tar.gz (1.6 MB)
      | 1.6 MB 5.5 MB/s
Requirement already satisfied: scipy in /usr/local/lib/python3.8/dist-
packages (from semopy) (1.7.3)
Requirement already satisfied: numpy in /usr/local/lib/python3.8/dist-packages
(from semopy) (1.21.6)
Requirement already satisfied: pandas in /usr/local/lib/python3.8/dist-packages
(from semopy) (1.3.5)
Requirement already satisfied: sympy in /usr/local/lib/python3.8/dist-packages
(from semopy) (1.7.1)
Collecting sklearn
    Downloading sklearn-0.0.post1.tar.gz (3.6 kB)
Requirement already satisfied: statsmodels in /usr/local/lib/python3.8/dist-
packages (from semopy) (0.12.2)
Requirement already satisfied: python-dateutil>=2.7.3 in
/usr/local/lib/python3.8/dist-packages (from pandas->semopy) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.8/dist-
packages (from pandas->semopy) (2022.6)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.8/dist-
packages (from python-dateutil>=2.7.3->pandas->semopy) (1.15.0)
Requirement already satisfied: patsy>=0.5 in /usr/local/lib/python3.8/dist-
packages (from statsmodels->semopy) (0.5.3)
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.8/dist-
packages (from sympy->semopy) (1.2.1)
Building wheels for collected packages: semopy, sklearn
  Building wheel for semopy (setup.py) ... done
    Created wheel for semopy: filename=semopy-2.3.9-py3-none-any.whl size=1657804
sha256=b635a09ca1791f54d3c80c4b6bd418039a87fc8b0e0ad9436e560e8ad81c1a33
  Stored in directory: /root/.cache/pip/wheels/aa/d5/83/afbfa4fe06d08c0ec7849e93
aa71843aa514684b3f22e3a694
  Building wheel for sklearn (setup.py) ... done
    Created wheel for sklearn: filename=sklearn-0.0.post1-py3-none-any.whl
size=2344
sha256=e6bd37648f212e93d3621e6b60fff4311c73ae1dc1688608a5382f28cc5e9e90
  Stored in directory: /root/.cache/pip/wheels/14/25/f7/1cc0956978ae479e75140219
088deb7a36f60459df242b1a72
Successfully built semopy sklearn
Installing collected packages: sklearn, semopy
Successfully installed semopy-2.3.9 sklearn-0.0.post1

```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting factor_analyzer
  Downloading factor_analyzer-0.4.1.tar.gz (41 kB)
    | 41 kB 593 kB/s
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
    Preparing wheel metadata ... done
Requirement already satisfied: numpy in /usr/local/lib/python3.8/dist-packages (from factor_analyzer) (1.21.6)
Collecting pre-commit
  Downloading pre_commit-2.20.0-py2.py3-none-any.whl (199 kB)
    | 199 kB 9.1 MB/s
Requirement already satisfied: scipy in /usr/local/lib/python3.8/dist-packages (from factor_analyzer) (1.7.3)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.8/dist-packages (from factor_analyzer) (1.0.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.8/dist-packages (from factor_analyzer) (1.3.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.8/dist-packages (from pandas->factor_analyzer) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.8/dist-packages (from pandas->factor_analyzer) (2022.6)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.8/dist-packages (from python-dateutil>=2.7.3->pandas->factor_analyzer) (1.15.0)
Collecting cfgv>=2.0.0
  Downloading cfgv-3.3.1-py2.py3-none-any.whl (7.3 kB)
Collecting nodeenv>=0.11.1
  Downloading nodeenv-1.7.0-py2.py3-none-any.whl (21 kB)
Collecting identify>=1.0.0
  Downloading identify-2.5.9-py2.py3-none-any.whl (98 kB)
    | 98 kB 8.0 MB/s
Requirement already satisfied: toml in /usr/local/lib/python3.8/dist-packages (from pre-commit->factor_analyzer) (0.10.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.8/dist-packages (from pre-commit->factor_analyzer) (6.0)
Collecting virtualenv>=20.0.8
  Downloading virtualenv-20.17.1-py3-none-any.whl (8.8 MB)
    | 8.8 MB 63.2 MB/s
Requirement already satisfied: setuptools in /usr/local/lib/python3.8/dist-packages (from nodeenv>=0.11.1->pre-commit->factor_analyzer) (57.4.0)
Requirement already satisfied: filelock<4,>=3.4.1 in /usr/local/lib/python3.8/dist-packages (from virtualenv>=20.0.8->pre-commit->factor_analyzer) (3.8.0)
Requirement already satisfied: platformdirs<3,>=2.4 in /usr/local/lib/python3.8/dist-packages (from virtualenv>=20.0.8->pre-commit->factor_analyzer) (2.5.4)
```

```

Collecting distlib<1,>=0.3.6
  Downloading distlib-0.3.6-py2.py3-none-any.whl (468 kB)
    | 468 kB 64.2 MB/s
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.8/dist-packages (from scikit-learn->factor_analyzer)
(3.1.0)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.8/dist-
packages (from scikit-learn->factor_analyzer) (1.2.0)
Building wheels for collected packages: factor-analyzer
  Building wheel for factor-analyzer (PEP 517) ... done
  Created wheel for factor-analyzer:
filename=factor_analyzer-0.4.1-py2.py3-none-any.whl size=42034
sha256=beb2cba594a7cac7b38d48ad77defdda84758a6b9a8f3ad396dd3c1a12aa6467
  Stored in directory: /root/.cache/pip/wheels/f5/8f/2e/a689c21bc4bf04f84ceebf4b
1f5846cacc04bfe179e7ad5ab0
Successfully built factor-analyzer
Installing collected packages: distlib, virtualenv, nodeenv, identify, cfgv,
pre-commit, factor-analyzer
Successfully installed cfgv-3.3.1 distlib-0.3.6 factor-analyzer-0.4.1
identify-2.5.9 nodeenv-1.7.0 pre-commit-2.20.0 virtualenv-20.17.1

```

```

[ ]: 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

from sklearn.preprocessing import OneHotEncoder

import warnings
warnings.filterwarnings("ignore")

%matplotlib inline

```

1 Pregunta 1

1. Cargue la base de datos y realice los ajustes necesarios para su uso (missing values, recodificar variables, etcetera). Identifique los tipos de datos que se encuentran en la base, realice estadísticas descriptivas sobre las variables importantes (Hint: Revisar la distribuciones, datos faltantes, outliers, etc.) y limpie las variables cuando sea necesario.

```
[ ]: try:  
    df = pd.read_csv("../data/junaeb2.csv")  
except:  
    try:  
        df = pd.read_csv("../..../data/junaeb2.csv")  
    except:  
        df = pd.read_csv("https://raw.githubusercontent.com/juancaros/LAB-MAA/  
        ↵main/data/junaeb2.csv")  
df.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
```

```
[ ]: junaeb = df.copy()
mean = junaeb.edad.mean()
std = junaeb.edad.std()
junaeb = junaeb[(junaeb.edad > mean - 3 * std) & (junaeb.edad < mean + 3 * std)]
junaeb = junaeb[junaeb.vive_padre != 2]
junaeb = junaeb[junaeb.vive_madre != 2]
junaeb["madre_trabaja"] = junaeb['madre_work'].apply(lambda x: 0 if x == 0 else 1)
junaeb["madre_empleada"] = junaeb['madre_work'].apply(lambda x: 1 if x == 1 else 0)
junaeb.dropna(inplace = True)
junaeb.reset_index(inplace = True)
```

Se realizó una limpieza de datos, la cual puede ser vista en mayor profundidad en el anexo. En primer lugar llama la atención la distribución etaria del estudio, en donde la mayoría pareciera ser de una mayor edad, resultado un promedio de edades de aproximadamente 82 años.

Además se encontraron algunas filas en vive_padre y vive_madre que fueron codificados como 2, dada la naturaleza binaria de estas variables estas filas fueron omitidas.

Se recodificó la variable madre_work, dado que es categorica, en 2 variables. La recodificación genera 2 nuevas variables: 1. madre_trabaja: indica si la madre trabaja o no (0 en caso de que sea desempleada, 1 en los otros 2 casos) 2. madre_empleada: indica si la madre es empleada (1 si es empleada, 0 si no trabaja o si realiza labores domésticas).

2 Pregunta 2

- Usando las variables sk1-sk13 realice un PCA. En particular, identifique los valores propios y determine el numero optimo de componentes. Luego estime y grafique la distribucion de los componentes. Ademas discuta la importancia relativa de las variables sobre cada uno de los componentes estimados. Que se puede concluir de este analisis?

```
[ ]: sk_vars = [f'sk{i+1}' for i in range(13)]
junaeb[sk_vars] = 6 - junaeb[sk_vars]
sk = junaeb[sk_vars]
```

Se transformaron las variables sk para que expresen el sentido contrario al original, inviertiendo la escala. Es decir, en vez de 1: Siempre y 5: Nunca, pasa a ser 5: Siempre y 1: Nunca.

Si bien el PCA está diseñado para valores continuos, se asumirá que no generará un efecto tan grande.

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

pca = PCA(n_components=len(sk.columns))
pca_features = pca.fit_transform(sk)
print(pca.explained_variance_ratio_) # Imprimir los factores propios

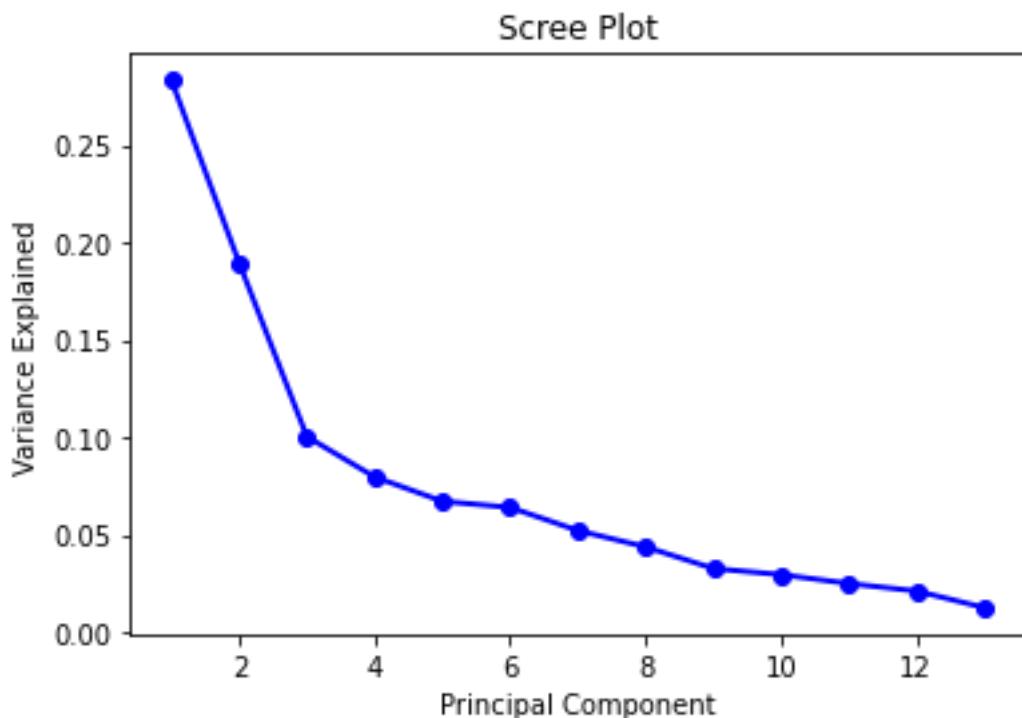
PC_values = np.arange(pca.n_components_) + 1
```

```

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()

```

```
[0.28325097 0.18947031 0.10030152 0.07959822 0.06710186 0.06395742
 0.05208095 0.04369425 0.03260041 0.02946809 0.02489866 0.02105562
 0.01252172]
```



Primero se realizó un scree plot para un PCA con número de factores igual al número de variables, en el cual se puede observar que el posible número óptimo de parámetros es 3, dado que ahí ocurre el punto de inflexión donde la cantidad de varianza explicada adicional es mínima para los demás componentes. SE decide volver a definir un PCA, pero ahora con 3 factores. Para los cuales los valores propios corresponde a 0.28325097, 0.18947031 y 0.10030152

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

```

pca = PCA(n_components=3)
pca_features = pca.fit_transform(sk)
print(pca.explained_variance_ratio_)

PC_values = np.arange(pca.n_components_) + 1

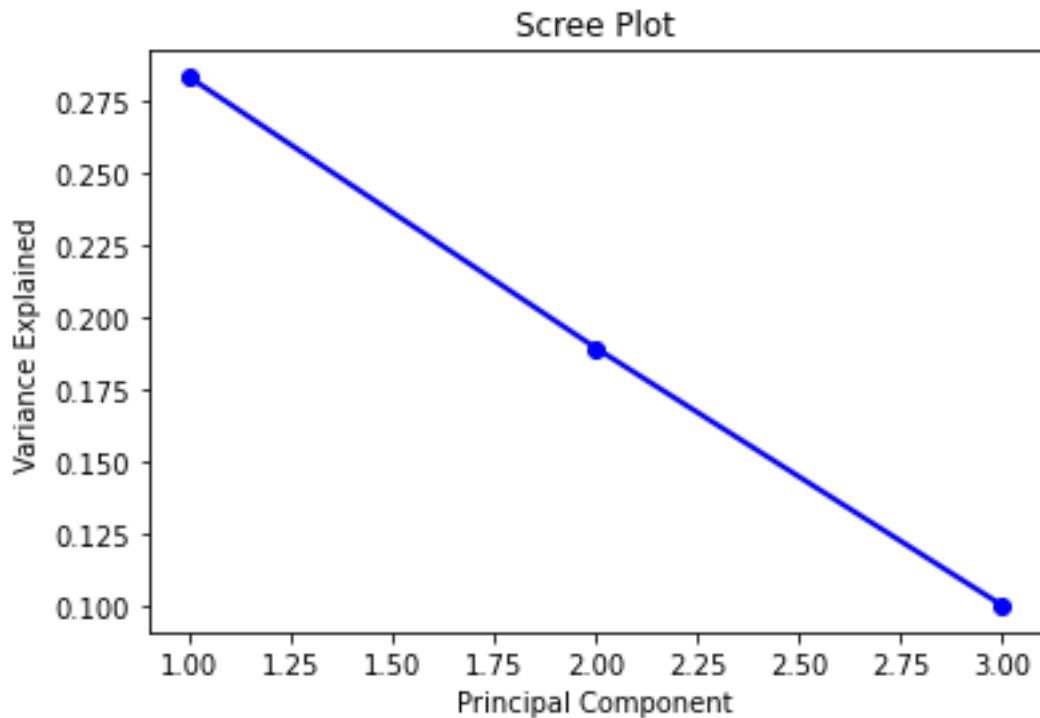
```

```

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()

```

[0.28325097 0.18947031 0.10030152]



[]: pca_vectors = pd.DataFrame(data = pca.components_)
pca_vectors.head()

[]:

	0	1	2	3	4	5	6	\
0	-0.102512	-0.228558	-0.168733	-0.183869	-0.183352	-0.251857	-0.352430	
1	-0.008483	-0.033357	-0.063066	-0.053761	-0.041009	0.028826	0.922524	
2	-0.082905	-0.268985	-0.182572	-0.189084	-0.295423	-0.331838	0.113989	
	7	8	9	10	11	12		
0	-0.292296	-0.236969	-0.403957	-0.285545	-0.291583	-0.427069		
1	-0.104449	-0.087411	-0.169113	-0.103665	-0.156056	-0.236927		
2	-0.483936	-0.079038	0.384483	0.000188	0.091548	0.494625		

Existen variables que no tienen una asociación fuerte con ningún factor, o tienen números similares para más de un factor, adicionalmente en el siguiente gráfico se puede ver que los factores no poseen

correlaciones entre sí.

```
[ ]: pca_df = pd.DataFrame(data=pca_features,columns= [f'PC{i+1}' for i in range(3)])
pca_df.corr().apply(lambda s: s.apply('{0:.3f}'.format))
```

```
[ ]:      PC1      PC2      PC3
PC1    1.000   -0.000   -0.000
PC2   -0.000    1.000    0.000
PC3   -0.000    0.000    1.000
```

```
[ ]: pca_scatter =pd.concat((pca_df, sk), axis = 1)
```

En los siguientes gráficos se puede ver la distribución de los componentes, a los que se les aplica un color dependiendo de cada una de las 13 variables.

PC1 y PC2: Comparando con la variable SK7 se ve que ambos contribuyen información, debido a que en el gráfico se pueden apreciar ciertas líneas horizontales pero, con cierta pendiente positiva, indicando que ambas variables influyen a la clasificación de las variables

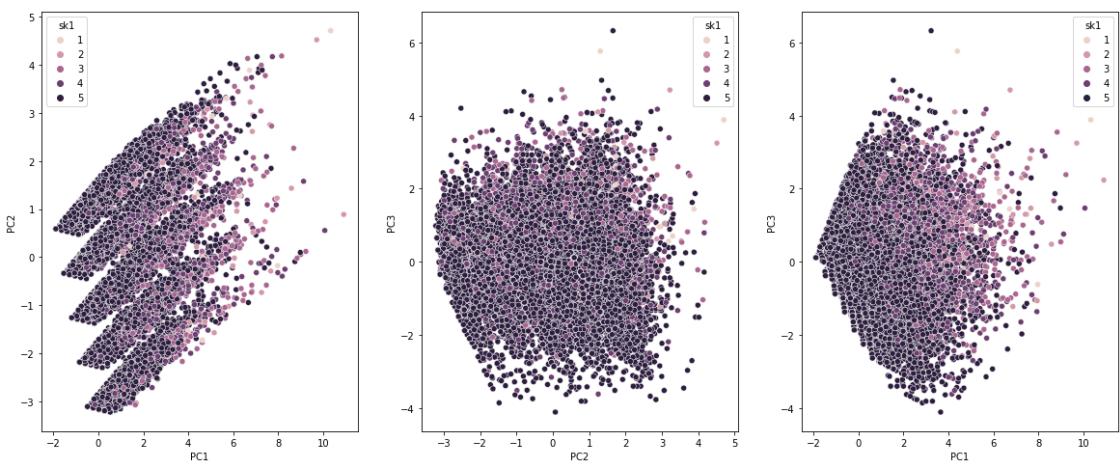
PC2 y PC3: Comparando con la variable SK7 se aprecia que para los valores positivos, se pueden ver ciertas líneas verticales indicando que la variable PC2 está entregando más información que PC3. Mientras que para los valores negativos es más notorio que ambas variables están entregando información (líneas pendiente negativa)

PC1 y PC3: se probó utilizando distintas variables pero el gráfico no indica una relación visual clara, pudiendo referirse a que no entregan mucha información ambas variables.

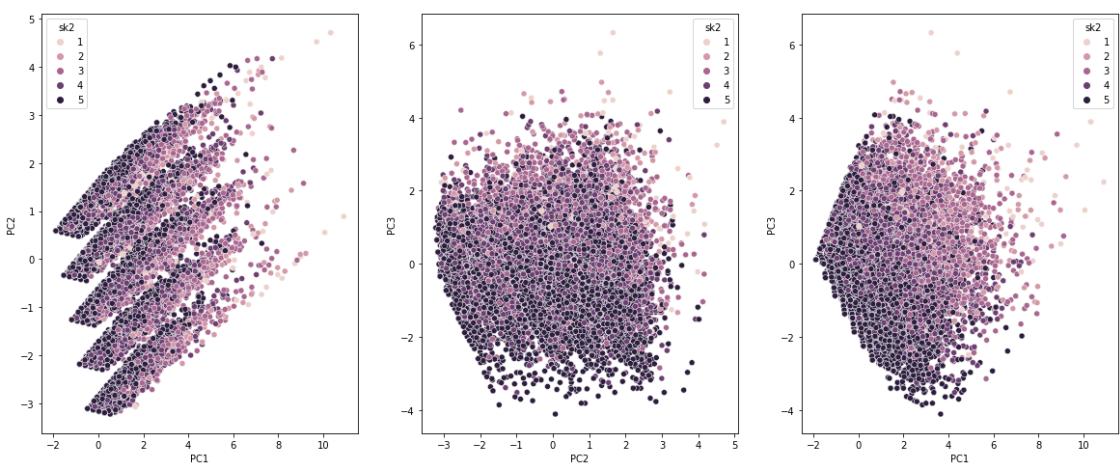
Para las variables sk2, sk5, sk6 (especialmente en el gráfico PC1 vs PC2), sk10 y sk13 se pueden diferenciar algunos grupos de color, aunque en forma menos evidente que las variables mencionadas anteriormente,

```
[ ]: for s in sk_vars:
    fig, ax = plt.subplots(1,3, figsize = (20,8))
    fig.suptitle(s)
    sns.scatterplot('PC1', 'PC2', data=pca_scatter, ax = ax[0], hue = s)
    sns.scatterplot('PC2', 'PC3', data=pca_scatter, ax = ax[1], hue = s)
    sns.scatterplot('PC1', 'PC3', data=pca_scatter, ax = ax[2], hue = s)
```

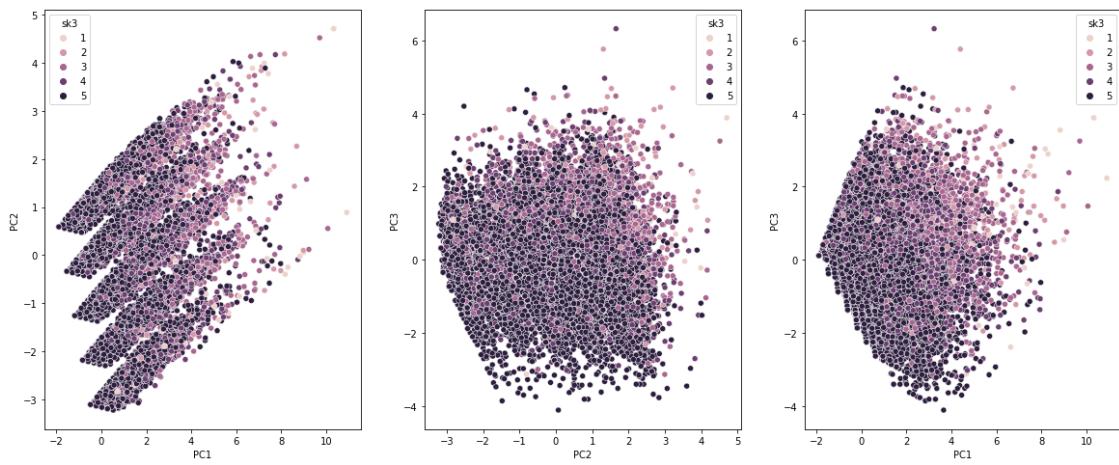
sk1



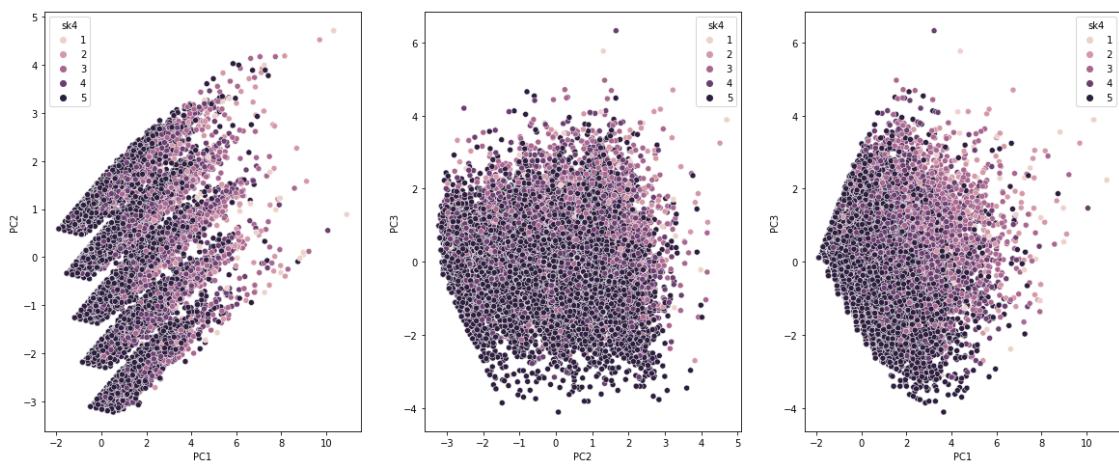
sk2



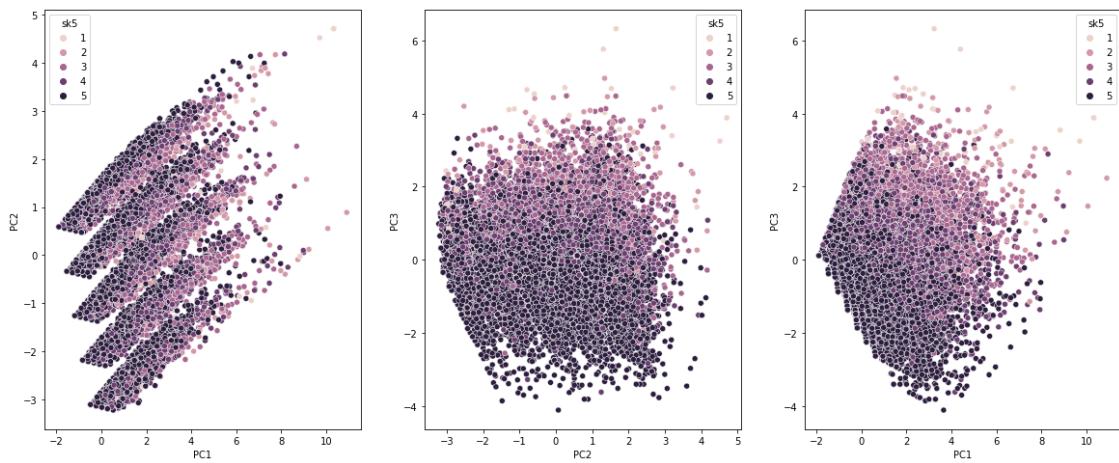
sk3



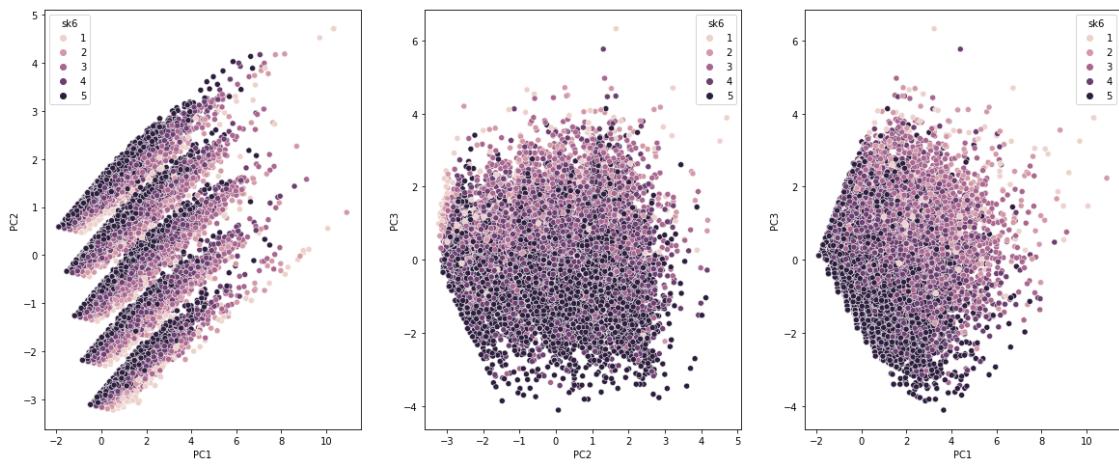
sk4



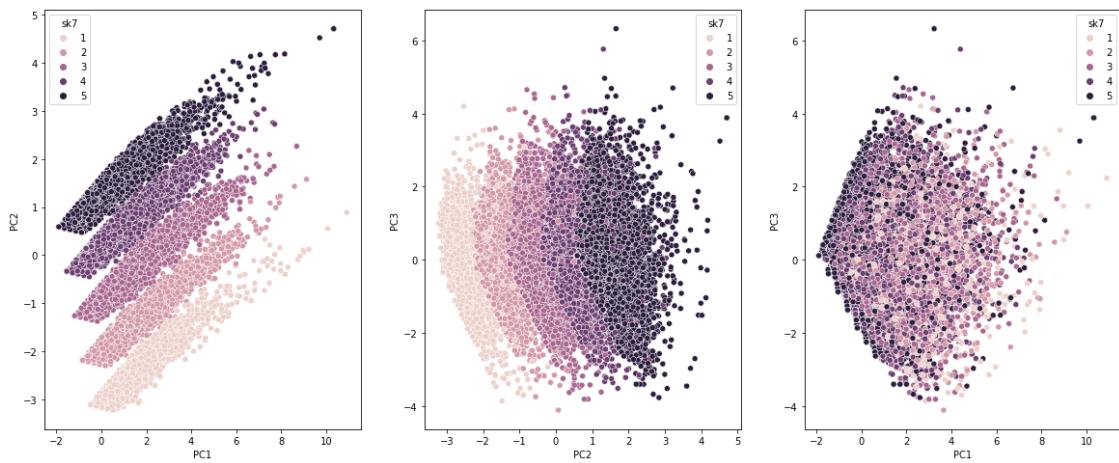
sk5



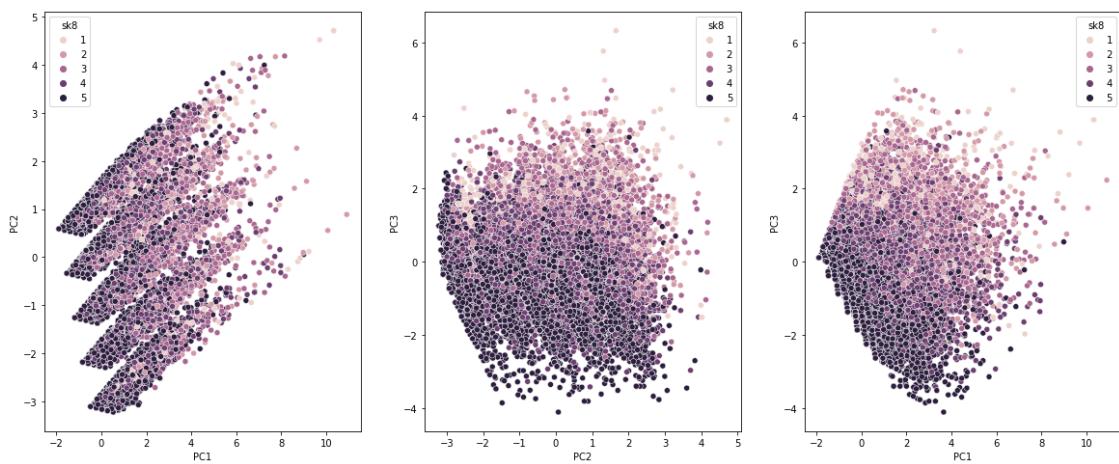
sk6



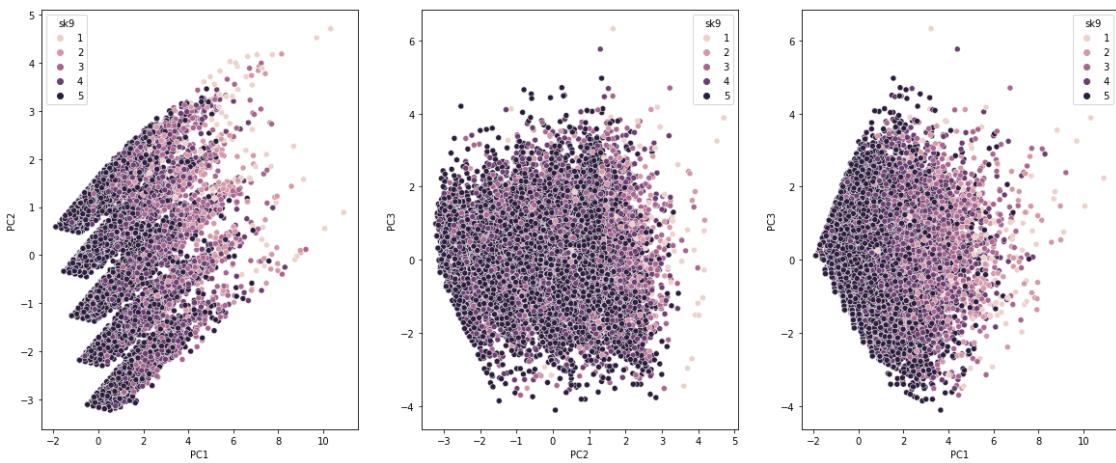
sk7



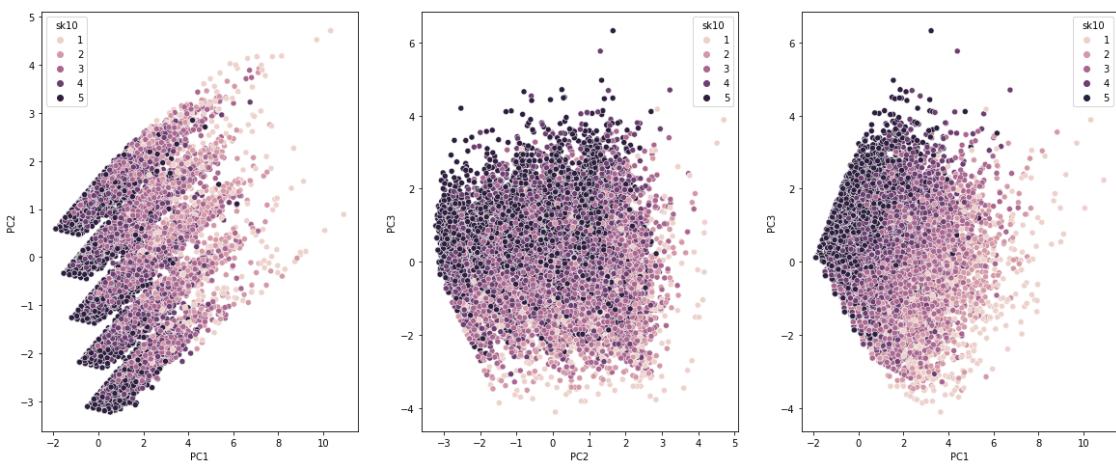
sk8



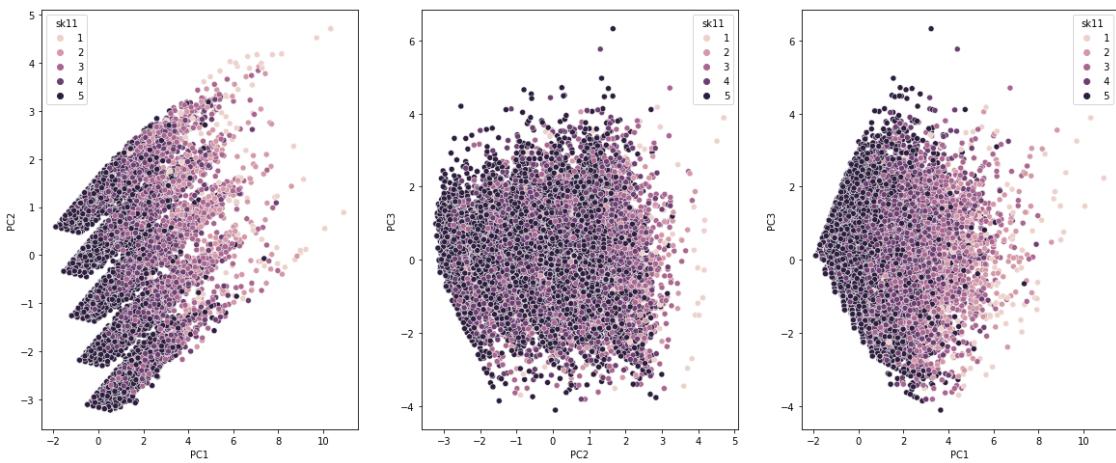
sk9



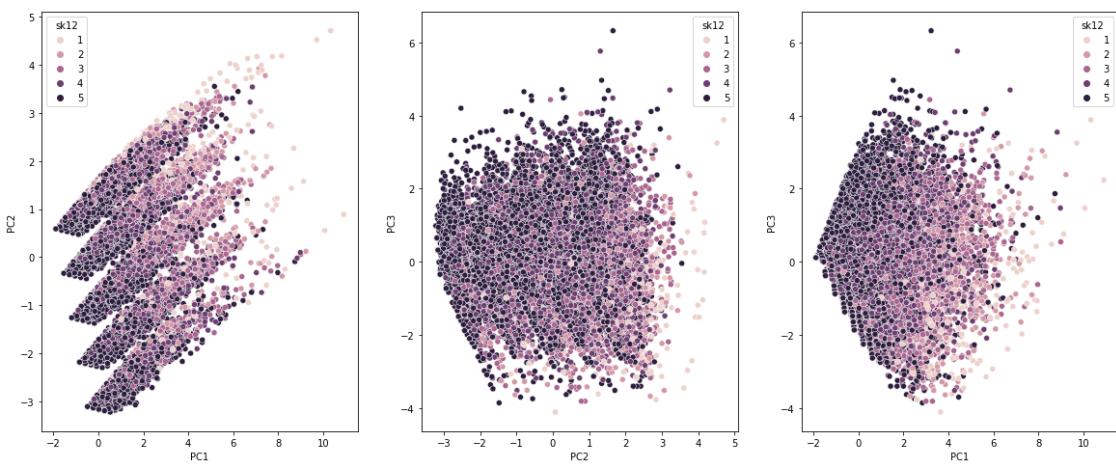
sk10

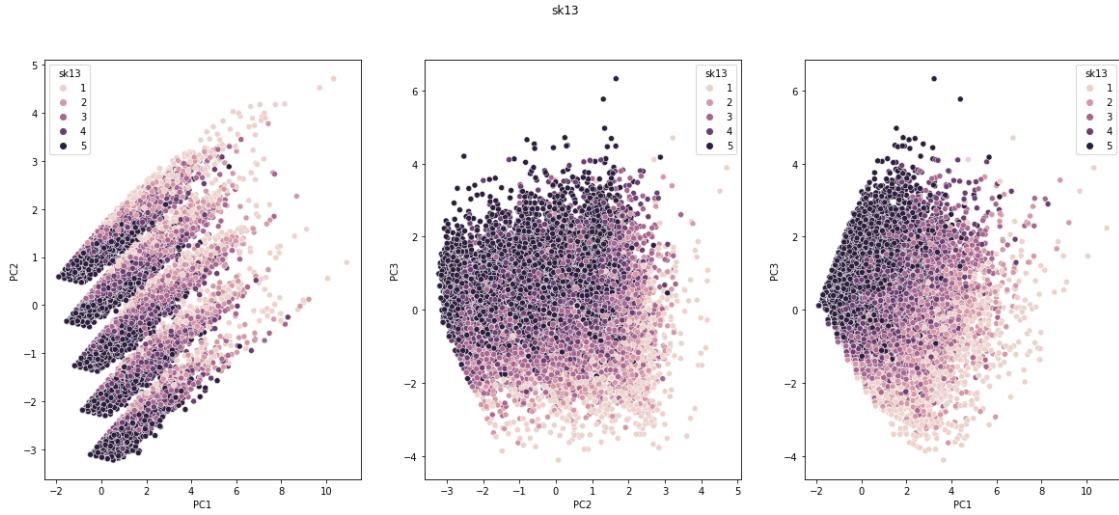


sk11



sk12





3 Pregunta 3

3. Con los resultados de la Pregunta 2, mantenga los primeros 3 componentes principales. Gráficamente indique si existen diferencias significativas entre grupos usando las siguientes variables: sexo, area, madre_work y act_fisica. Que puede concluir de los resultados?

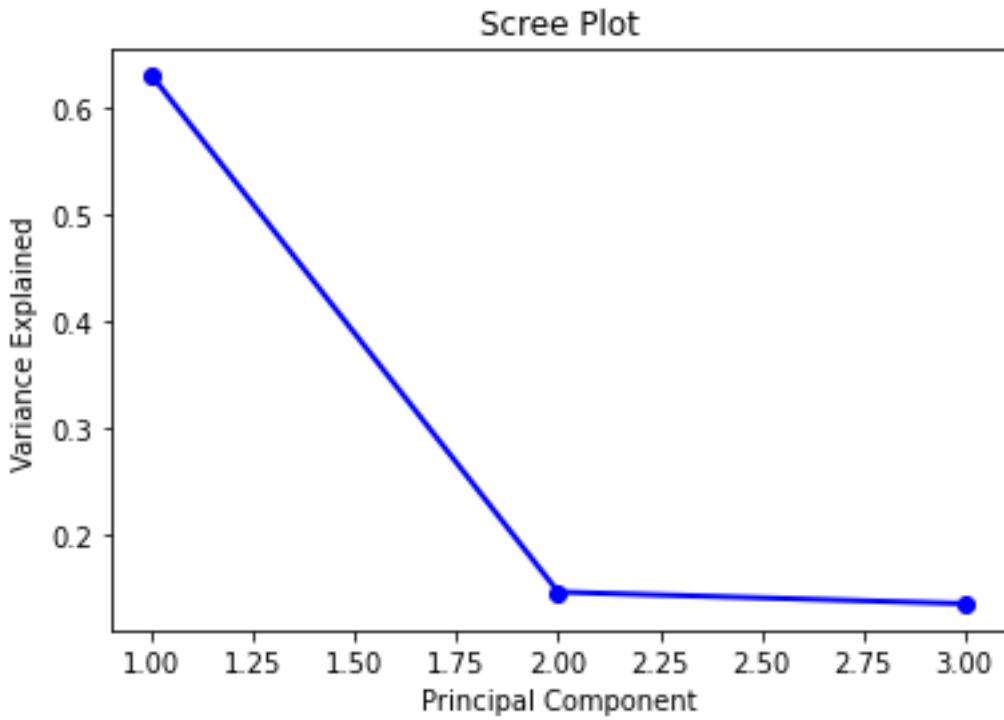
```
[ ]: junaeb2 = junaeb[['sexo', 'area', 'madre_trabaja', ↪
    'madre_empleada', 'act_fisica']] # madre_trabaja, madre_empleada 'madre_work'
```

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

```
pca = PCA(n_components=3)
pca_features = pca.fit_transform(junaeb2)
print(pca.explained_variance_ratio_)

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()
```

```
[0.62959487 0.14675232 0.13589239]
```



```
[ ]: pca_vectors = pd.DataFrame(data = pca.components_)
pca_vectors.head()
```

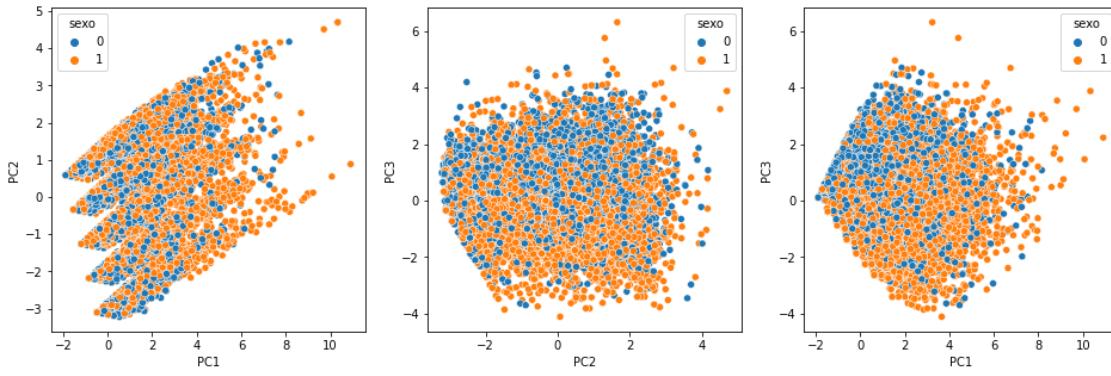
```
[ ]:          0         1         2         3         4
0  0.028420 -0.021011  0.003504 -0.009230  0.999326
1  0.197651 -0.070726 -0.279349 -0.936844 -0.014782
2 -0.979859 -0.015490 -0.057970 -0.188681  0.026001
```

Alta relación entre PC3 y Sexo, PC2 y madre trabaja, PC1 y madre empleada.

A continuación se presentan las diferencias significativas presentadas para cada una de las variables utilizadas. En este caso se pueden diferenciar grupos rápidamente para todos los casos.

```
[ ]: pca_scatter = pd.concat((pca_df, junaeb2), axis = 1)
fig, ax = plt.subplots(1,3, figsize = (16,5))
h = 'sexo'
sns.scatterplot('PC1', 'PC2', data=pca_scatter, ax = ax[0], hue = h)
sns.scatterplot('PC2', 'PC3', data=pca_scatter, ax = ax[1], hue = h)
sns.scatterplot('PC1', 'PC3', data=pca_scatter, ax = ax[2], hue = h)
```

```
[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbedf737c70>
```

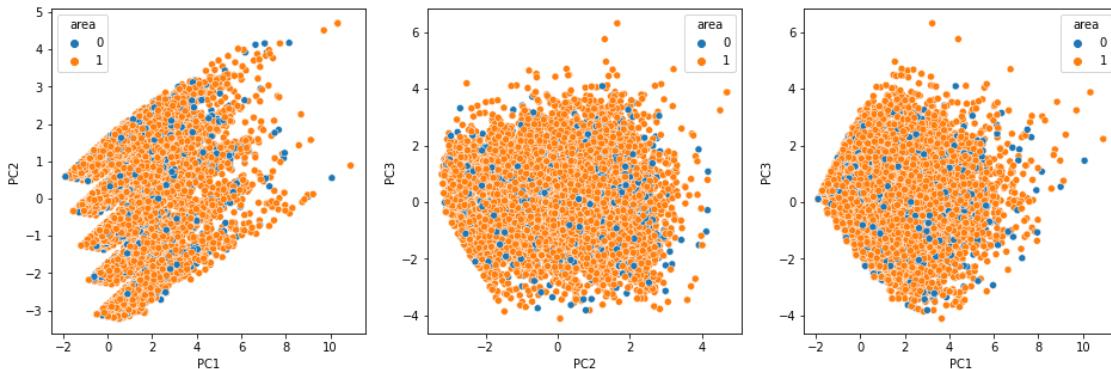


A Partir del grafico, la relación entre PC1 y PC2, no hay una separación clara de los resultados no habiendo una relación clara

Para PC3-PC2 y PC1-PC3 se ve una diferenciación clara filtrando por sexo, creando una “línea” horizontal entre ellas. Para PC1 - PC2 se ven grupos pero cuya definicion no es tan clara como los anteriores, en que se podría trazar facilmente una recta que separara ambos grupos

```
[ ]: pca_scatter =pd.concat((pca_df, junaeb2), axis = 1)
fig, ax = plt.subplots(1,3, figsize = (16,5))
h = 'area'
sns.scatterplot('PC1', 'PC2', data=pca_scatter, ax = ax[0], hue = h)
sns.scatterplot('PC2', 'PC3', data=pca_scatter, ax = ax[1], hue = h)
sns.scatterplot('PC1', 'PC3', data=pca_scatter, ax = ax[2], hue = h)
```

```
[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbef6c4310>
```



Para el caso de la variable área, se ven diferencias en los gráficos para todas las combinaciones de factores, sin embargo están no son tan claras como las de la variable sexo

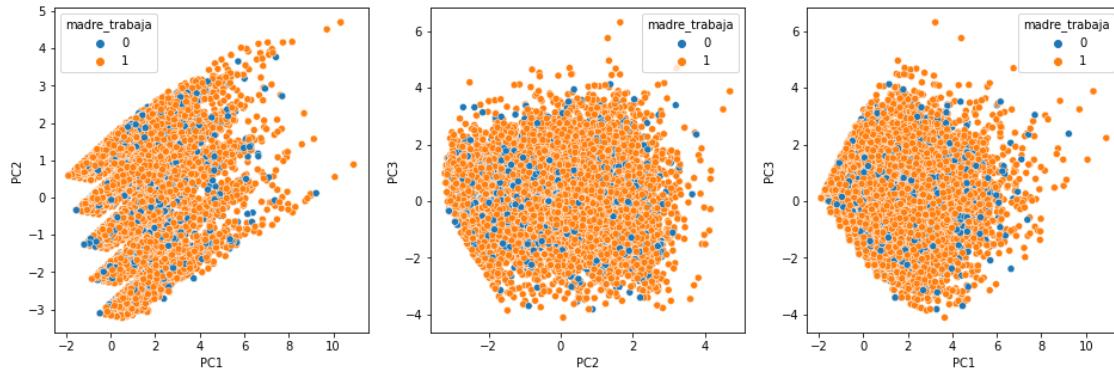
```
[ ]: pca_scatter =pd.concat((pca_df, junaeb2), axis = 1)
fig, ax = plt.subplots(1,3, figsize = (16,5))
```

```

h = 'madre_trabaja'
sns.scatterplot('PC1', 'PC2', data=pca_scatter, ax = ax[0], hue = h)
sns.scatterplot('PC2', 'PC3', data=pca_scatter, ax = ax[1], hue = h)
sns.scatterplot('PC1', 'PC3', data=pca_scatter, ax = ax[2], hue = h)

```

[]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbedf614940>



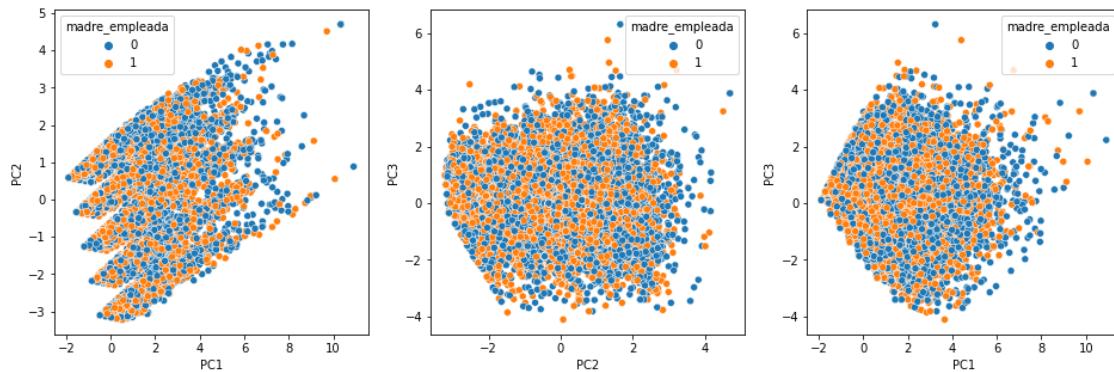
En este caso para las combinaciones PC1-PC2 y PC2-PC3 se ven diferencias claras en el eje correspondiente a PC2, lo que insinua que este factor es el más relacionado a la variable madre_trabaja (Proveniente de MAdre Work)

```

pca_scatter = pd.concat((pca_df, junaeb2), axis = 1)
fig, ax = plt.subplots(1,3, figsize = (16,5))
h = 'madre_empleada'
sns.scatterplot('PC1', 'PC2', data=pca_scatter, ax = ax[0], hue = h)
sns.scatterplot('PC2', 'PC3', data=pca_scatter, ax = ax[1], hue = h)
sns.scatterplot('PC1', 'PC3', data=pca_scatter, ax = ax[2], hue = h)

```

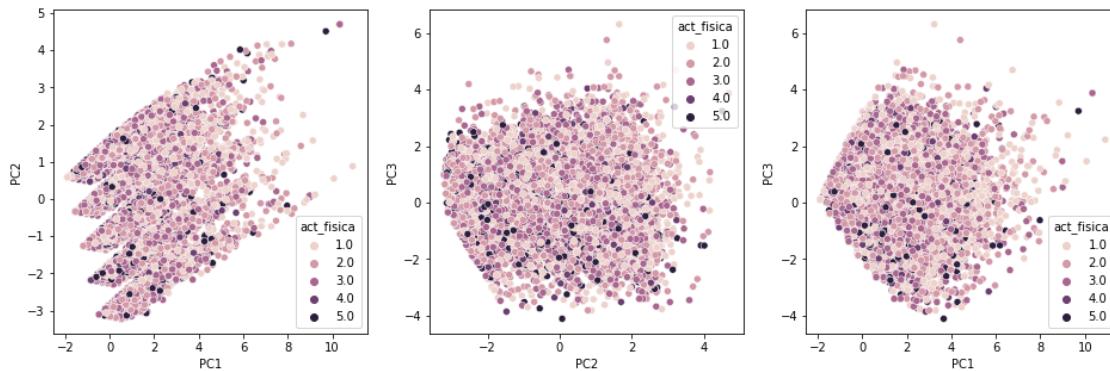
[]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbedffd0f70>



Para la variable madre_empleada PC1-PC2 y PC2-PC3 se ven diferencias claras en el eje correspondiente a PC2 nuevamente, lo que insinua que este factor es el más relacionado a la variable.

```
[ ]: pca_scatter = pd.concat((pca_df, junaeb2), axis = 1)
fig, ax = plt.subplots(1,3, figsize = (16,5))
h = 'act_fisica'
sns.scatterplot('PC1', 'PC2', data=pca_scatter, ax = ax[0], hue = h)
sns.scatterplot('PC2', 'PC3', data=pca_scatter, ax = ax[1], hue = h)
sns.scatterplot('PC1', 'PC3', data=pca_scatter, ax = ax[2], hue = h)
```

```
[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbee35afdc0>
```



Finalmente al graficar las combinaciones de factores, asignando color según la act_fisica asociada, se pueden ver diferencias de grupos en los ejes que incluyen al factor PC1, lo que indica que este se relaciona con la variable act_fisica.

4 Pregunta 4

4. A partir del mismo set de variables sk1-sk13 realice un EFA. En particular determine el numero optimo de factores y las variables que se asocian a cada factor. Tambien discuta si existen variables que no son informativas (Hint: para realizar un EFA, todas las variables deben estar representadas en el mismo sentido logico. Si una caracteristica es negativa debe ser invertida en la escala, de tal forma que todas las variables representen aspectos positivos).

```
[ ]: # Create factor analysis object and perform factor analysis
fa = FactorAnalyzer(rotation='promax')
fa.fit(sk)
```

```
[ ]: FactorAnalyzer(rotation_kwarg={})
```

```
[ ]: fa.loadings_
```

```
[ ]: array([[ 0.01364962,  0.60268962, -0.03425554],
       [-0.03642904,  0.49078497,  0.23392701],
```

```
[ 0.02709609,  0.64435192, -0.04501677],
[ 0.00425373,  0.73875116, -0.03567302],
[-0.14432621, -0.02896221,  0.85824447],
[ 0.00703791,  0.04023962,  0.51404974],
[ 0.0123327 ,  0.04133763,  0.14001644],
[ 0.14943376, -0.10910951,  0.50966003],
[ 0.47937536,  0.08418939,  0.04971678],
[ 0.61756808, -0.03742814, -0.03088626],
[ 0.69295865,  0.03221508,  0.00410174],
[ 0.56651891, -0.01862305, -0.00774673],
[ 0.52366539,  0.0156779 , -0.007063 ]])
```

```
[ ]: pd.DataFrame(fa.loadings_, columns = ['PC1', 'PC2', 'PC3'], index = sk_vars)
```

```
[ ]:      PC1      PC2      PC3
sk1  0.013650  0.602690 -0.034256
sk2 -0.036429  0.490785  0.233927
sk3  0.027096  0.644352 -0.045017
sk4  0.004254  0.738751 -0.035673
sk5 -0.144326 -0.028962  0.858244
sk6  0.007038  0.040240  0.514050
sk7  0.012333  0.041338  0.140016
sk8  0.149434 -0.109110  0.509660
sk9  0.479375  0.084189  0.049717
sk10 0.617568 -0.037428 -0.030886
sk11 0.692959  0.032215  0.004102
sk12 0.566519 -0.018623 -0.007747
sk13 0.523665  0.015678 -0.007063
```

A partir del metodo pro max se estiman pesos relativos y el numero de factores resulta ser 3. Aquí salta la atención respecto a que tan informativa es la variable sk7, ya que no se asocia en forma fuerte a algun factor en específico.

```
[ ]: fa.get_eigenvalues() #original eigenvalues y common factor eigenvalues
```

```
[ ]: (array([3.97194099, 1.3606774 , 1.10573516, 0.99945812, 0.78767797,
       0.71753445, 0.70951584, 0.69900009, 0.61413939, 0.53317744,
       0.52112931, 0.5025337 , 0.47748014]),
array([ 3.37373677,  0.76582593,  0.59033664,  0.20711582,  0.08703898,
       0.06286435,  0.035794 ,  0.02747806, -0.02400405, -0.07755832,
      -0.09409038, -0.11231338, -0.17587107]))
```

Utilizando semopy, la aproximación de la representación del modelo resulta ser en 4 factores latentes vs los 3 factores obtenidos en promax. En el output de la siguiente celda además se puede ver como las variables se distribuyen por los distintos factores.

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

```

eta1 =~ sk11 + sk9 + sk10 + sk12
eta2 =~ sk6 + sk7
eta3 =~ sk4 + sk2 + sk11 + sk5 + sk3 + sk9 + sk1 + sk8 + sk12
eta4 =~ sk11 + sk12 + sk13

```

5 Pregunta 5

- Con los resultados obtenidos en la Pregunta 4, proponga un CFA donde cada variable solo se asocia con un factor. Entregue un nombre a cada factor que representa el concepto comun entre todas las variables. Reporte la importancia de cada medida (variable) a cada factor e indique la correlacion entre factores.

Se utilizan 3 factores llamados Habilidades de Descubrimiento y Creación(descubrimiento), Habilidades Afectivas (afectivas) y Habilidades de Socialización (socializacion), esto después de haber clasificado los elementos de la tabla de factores para la estiamcion promax de la pregunta anterior y ver si existe algún nombre que agrupe a los factores. Estos factores se agrupan en la siguiente forma:

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

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

Habilidades de Socialización - sk5: juega con otros (1: nunca - 5: siempre) - sk6: comparte sus cosas con otros (1: nunca - 5: siempre) - sk8: participa en juegos grupales (1: nunca - 5: siempre)

Finalmente el orden de las variables en el modelo se deteminó rankeando como afectan estas a cada factor, dado que los pesos se determinan tomandola como referencia.

```

[ ]: mod = """
# measurement model
descubrimiento =~ sk11 + sk9 + sk10 + sk12 + sk13
afectivas =~ sk4 + sk1 + sk2 + sk3
socializacion =~ sk5 + sk6 + sk8
"""

model = semopy.Model(mod)
out=model.fit(sk)
print(out)

```

```

Name of objective: MLW
Optimization method: SLSQP
Optimization successful.
Optimization terminated successfully
Objective value: 0.140

```

```

Number of iterations: 32
Params: 0.795 1.094 0.900 1.058 0.554 1.065 0.915 1.020 1.147 0.242 0.202 0.195
0.273 0.694 0.589 0.365 0.159 0.429 0.149 0.090 0.478 0.151 0.108 0.100 0.222
0.107 0.156

```

Una vez estimado el modelo se evalua su ajuste, observando que todas las relaciones planteadas entre factores y variables son significativas.

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

	lval	op	rval	Estimate	Est.	Std	Std. Err	\
0	sk11	~	descubrimiento	1.000000	0.723489		-	
1	sk9	~	descubrimiento	0.794963	0.582444	0.007079		
2	sk10	~	descubrimiento	1.094109	0.557499	0.010099		
3	sk12	~	descubrimiento	0.900459	0.543437	0.008492		
4	sk13	~	descubrimiento	1.057554	0.513197	0.010478		
5	sk4	~	afectivas	1.000000	0.697841		-	
6	sk1	~	afectivas	0.553990	0.582797	0.004948		
7	sk2	~	afectivas	1.064869	0.644010	0.008814		
8	sk3	~	afectivas	0.915012	0.627150	0.00772		
9	sk5	~	socializacion	1.000000	0.715107		-	
10	sk6	~	socializacion	1.020033	0.554582	0.010598		
11	sk8	~	socializacion	1.146704	0.547862	0.012007		
12	afectivas	~~	afectivas	0.151130	1.000000	0.001844		
13	afectivas	~~	descubrimiento	0.108097	0.590388	0.001314		
14	afectivas	~~	socializacion	0.099826	0.650428	0.001182		
15	descubrimiento	~~	descubrimiento	0.221820	1.000000	0.002615		
16	descubrimiento	~~	socializacion	0.106894	0.574891	0.001357		
17	socializacion	~~	socializacion	0.155861	1.000000	0.002052		
18	sk2	~~	sk2	0.241824	0.585251	0.001835		
19	sk11	~~	sk11	0.201957	0.476564	0.001827		
20	sk3	~~	sk3	0.195175	0.606683	0.001447		
21	sk9	~~	sk9	0.273042	0.660759	0.001939		
22	sk13	~~	sk13	0.693885	0.736629	0.004656		
23	sk10	~~	sk10	0.588810	0.689194	0.004086		
24	sk6	~~	sk6	0.365103	0.692439	0.002671		
25	sk4	~~	sk4	0.159210	0.513019	0.001331		
26	sk12	~~	sk12	0.429162	0.704677	0.002944		
27	sk5	~~	sk5	0.148925	0.488622	0.001574		
28	sk1	~~	sk1	0.090176	0.660347	0.000636		
29	sk8	~~	sk8	0.477858	0.699847	0.003468		
				z-value	p-value			
0		-	-					
1	112.29294		0.0					
2	108.338009		0.0					
3	106.034615		0.0					
4	100.930772		0.0					

```

5      -      -
6  111.964635  0.0
7  120.815923  0.0
8  118.521014  0.0
9      -      -
10  96.252221  0.0
11  95.50193  0.0
12  81.953841  0.0
13  82.246763  0.0
14  84.433063  0.0
15  84.82727  0.0
16  78.767626  0.0
17  75.962398  0.0
18  131.805105 0.0
19  110.547791 0.0
20  134.901406 0.0
21  140.815089 0.0
22  149.018677 0.0
23  144.117345 0.0
24  136.684736 0.0
25  119.618784 0.0
26  145.795142 0.0
27  94.63517  0.0
28  141.745112 0.0
29  137.801355 0.0

```

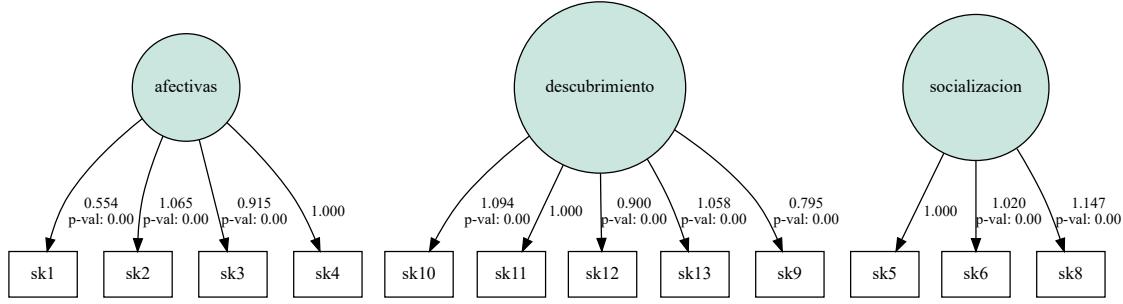
```
[ ]: semopy.calc_stats(model)
```

	DoF	DoF	Baseline	chi2	chi2	p-value	chi2	Baseline	CFI	\
Value	51		66	7830.430197		0.0	147409.547889	0.947202		
	GFI	AGFI	NFI	TLI	RMSEA		AIC	BIC		\
Value	0.94688	0.931256	0.94688	0.931673	0.052192	53.720337	294.913743			
	LogLik									
Value	0.139832									

Además, se encuentra un error cuadrático bastante bajo, indicando que el modelo estimado está bien ajustado. Finalmente, se muestra gráficamente el modelo.

```
[ ]: semopy.semplot(model, "model.png")
```

```
[ ]:
```



6 Pregunta 6

6. Finalmente, implemente un SEM completo usando la estructura propuesta en la Pregunta 5. En particular, estime un modelo donde los factores explican el nivel de actividad física, junto con otras variables que existen en la base de datos. Ademas utilice otras variables relevantes de la base de datos para explicar los factores latentes. Las variables a incluir en el modelo final deben tener sustento teorico y el modelo final debe optimizar el ajuste a los datos, en base a los criterios vistos en clase. Que puede concluir en base a sus resultados?

Se tomó el modelo anteriormente propuesto y se le agregaron algunas variables, las cuales corresponde a la siguientes:

- Factor de habilidades de descubrimiento:
 - *vive_madre*: fue añadida junto a *vive_padre* bajo el argumento de que la presencia de los padres pueden favorecer al desarrollo de las habilidades de descubrimiento de los niños y niñas mediante por ejemplo juegos, sin embargo *vive_padre* no fue significativo por lo que se eliminó del modelo.
 - *area*: se añadió la cercanía a las áreas urbanas a rurales de los niños bajo el pretexto de que el ambiente en el que se desarrollen puede brindarles más opciones de descubrimiento a los infantes.

- Factor de habilidades de habilidades afectivas:
 - *madre_trabaja* y *madre_empleada*: se agregaron pensando en que si bien los niños y niñas pueden vivir con sus madres, el si es que estas se involucran en la crianza durante todo el día o tienen un empleo que les obligue a salir del hogar puede generar diferencias.
- Factor de habilidades sociales:
 - *vive_padre*: se añadió junto a *vive_madre* bajo la idea de que la presencia de los padres afecta a los niños y niñas en el desarrollo de las habilidades sociales, al ser las primeras personas con las que se suelen relacionar en la vida. Sin embargo *vive_madre* resultó no ser significativo por lo que se sacó del modelo.
 - *edad*: tomando la idea de que las habilidades sociales mejoran a medida que los individuos crecen.

Luego se establecieron como regresiones propuestas explicar la *act_fisica* como la suma de los 3 factores descritos en los modelos anteriores, a los que además se les añade el área y la edad como factores explicativos.

Finalmente se le indica al modelo la existencia de varias correlaciones basadas en el heatmap presente en el anexo 2

```
[ ]: mod = """
# measurement model
desc =~ sk9 + sk10 + sk11 + sk12 + sk13 + area + vive_madre
affect =~ sk1 + sk2 + sk3 + sk4 + madre_trabaja + madre_empleada
social =~ sk5 + sk6 + sk8 + edad + vive_padre

# regressions
act_fisica ~ desc + afect + social + area + edad

# residual correlations
vive_padre ~~ sk1
vive_madre ~~ sk1
vive_madre ~~ madre_trabaja
vive_madre ~~ madre_empleada
madre_trabaja ~~ madre_empleada
sk3 ~~ sk4
sk5 ~~ sk8
sk9 ~~ sk11
"""

# imce ~ desc + afect + social + act_fisica

mod = semopy.Model(mod)
out=mod.fit(junaeb)
print(out)
```

Name of objective: MLW
 Optimization method: SLSQP
 Optimization successful.
 Optimization terminated successfully
 Objective value: 0.187
 Number of iterations: 72
 Params: 1.482 1.288 1.213 1.443 0.023 0.018 1.996 1.407 1.578 0.043 0.105 1.118
 1.148 -0.880 0.055 0.250 -0.527 0.616 -0.270 0.003 0.004 0.200 0.001 0.002 0.004
 0.024 0.052 0.093 0.050 0.224 0.031 0.168 0.023 0.288 0.215 0.578 0.357 0.424
 0.087 0.217 0.680 0.249 0.188 1.092 8.525 0.084 0.503 0.049 0.047 0.058 0.126
 0.080 0.137

```
[ ]: mod.inspect()
```

	lval	op	rval	Estimate	Std. Err	z-value	\
0	area	~	desc	0.023453	0.00398	5.893293	
1	edad	~	social	-0.880044	0.040537	-21.709782	
2	sk9	~	desc	1.000000	-	-	
3	sk10	~	desc	1.482128	0.016815	88.141695	
4	sk11	~	desc	1.288377	0.011826	108.943555	
5	sk12	~	desc	1.212626	0.013987	86.694551	

6		sk13	~	desc	1.442886	0.017056	84.59792
7	vive_madre	~		desc	0.018070	0.002114	8.547393
8		sk1	~	afect	1.000000	-	-
9		sk2	~	afect	1.996021	0.018235	109.461483
10		sk3	~	afect	1.406788	0.015044	93.514375
11		sk4	~	afect	1.577610	0.015317	102.999683
12	madre_trabaja	~		afect	0.043316	0.006687	6.477488
13	madre_empleada	~		afect	0.104800	0.010978	9.546216
14		sk5	~	social	1.000000	-	-
15		sk6	~	social	1.117687	0.012663	88.265912
16		sk8	~	social	1.147870	0.012288	93.411562
17	vive_padre	~		social	0.054947	0.006139	8.950468
18	act_fisica	~		desc	0.249667	0.02341	10.66518
19	act_fisica	~		afect	-0.526678	0.047862	-11.004097
20	act_fisica	~		social	0.616483	0.032	19.26522
21	act_fisica	~		area	-0.269657	0.015372	-17.542276
22	act_fisica	~		edad	0.003175	0.001535	2.068645
23		edad	~~	edad	8.525199	0.051221	166.4391
24		area	~~	area	0.083963	0.000502	167.285844
25		afect	~~	afect	0.049318	0.000735	67.105841
26		afect	~~	desc	0.047370	0.000683	69.309944
27		afect	~~	social	0.058124	0.000735	79.059961
28		desc	~~	desc	0.125699	0.002194	57.295382
29		desc	~~	social	0.079595	0.001136	70.048076
30		social	~~	social	0.136632	0.00211	64.747165
31	vive_padre	~~	sk1	0.003735	0.000605	6.171086	
32	vive_padre	~~	vive_padre	0.200468	0.001199	167.182096	
33	vive_madre	~~	sk1	0.000961	0.000207	4.645052	
34	vive_madre	~~	madre_trabaja	0.001565	0.000198	7.906295	
35	vive_madre	~~	madre_empleada	0.003685	0.000325	11.339075	
36	vive_madre	~~	vive_madre	0.023637	0.000141	167.236221	
37	madre_trabaja	~~	madre_empleada	0.051637	0.000679	76.033923	
38	madre_trabaja	~~	madre_trabaja	0.092731	0.000554	167.275501	
39		sk3	~~	sk4	0.050045	0.001188	42.131654
40		sk3	~~	sk3	0.224300	0.001624	138.13998
41		sk5	~~	sk8	0.030517	0.002045	14.924639
42		sk5	~~	sk5	0.168208	0.001776	94.695792
43		sk9	~~	sk11	0.023134	0.001562	14.813142
44		sk9	~~	sk9	0.287575	0.002149	133.787009
45		sk11	~~	sk11	0.215215	0.002019	106.585385
46		sk10	~~	sk10	0.578253	0.004123	140.264606
47		sk6	~~	sk6	0.356589	0.002731	130.568683
48		sk12	~~	sk12	0.424270	0.002972	142.753419
49		sk1	~~	sk1	0.087267	0.000647	134.979286
50		sk2	~~	sk2	0.216624	0.001891	114.576497
51		sk13	~~	sk13	0.679882	0.004665	145.737059
52	madre_empleada	~~	madre_empleada	0.249425	0.001492	167.210846	

53		sk4	~~	sk4	0.187743	0.001484	126.488994
54	act_fisica	~~	act_fisica	1.091991	0.006785	160.952279	
55		sk8	~~	sk8	0.502791	0.00394	127.62594
p-value							
0		0.0					
1		0.0					
2		-					
3		0.0					
4		0.0					
5		0.0					
6		0.0					
7		0.0					
8		-					
9		0.0					
10		0.0					
11		0.0					
12		0.0					
13		0.0					
14		-					
15		0.0					
16		0.0					
17		0.0					
18		0.0					
19		0.0					
20		0.0					
21		0.0					
22	0.038579						
23		0.0					
24		0.0					
25		0.0					
26		0.0					
27		0.0					
28		0.0					
29		0.0					
30		0.0					
31		0.0					
32		0.0					
33	0.000003						
34		0.0					
35		0.0					
36		0.0					
37		0.0					
38		0.0					
39		0.0					
40		0.0					
41		0.0					

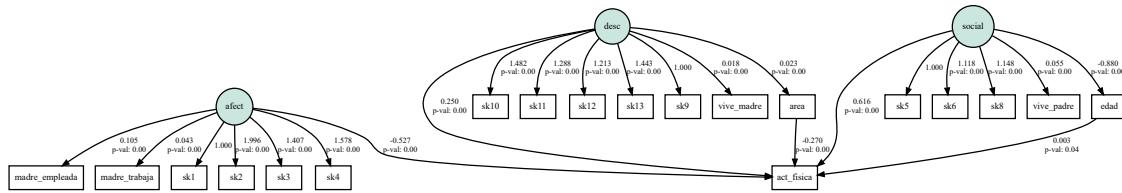
```

42      0.0
43      0.0
44      0.0
45      0.0
46      0.0
47      0.0
48      0.0
49      0.0
50      0.0
51      0.0
52      0.0
53      0.0
54      0.0
55      0.0

```

```
[ ]: semopy.semplot(mod, "semmodel.png")
```

```
[ ]:
```



Del ajuste del modelo se puede concluir que existe la posibilidad de generar factores a partir de las relaciones propuestas anteriormente, las cuales son significativas y que mediante la combinación de estos factores, en adición con la edad y el area se puede estimar el nivel de actividad física de los individuos en la base de datos de estudio junaeb. Cabe destacar que las relaciones son significativas con un 99% de confianza, exceptuando la relación entre la edad y la actividad física, la cual es significativa al 95%.

7 Anexo 1: Análisis de distribuciones

```
[ ]: df.info()
```

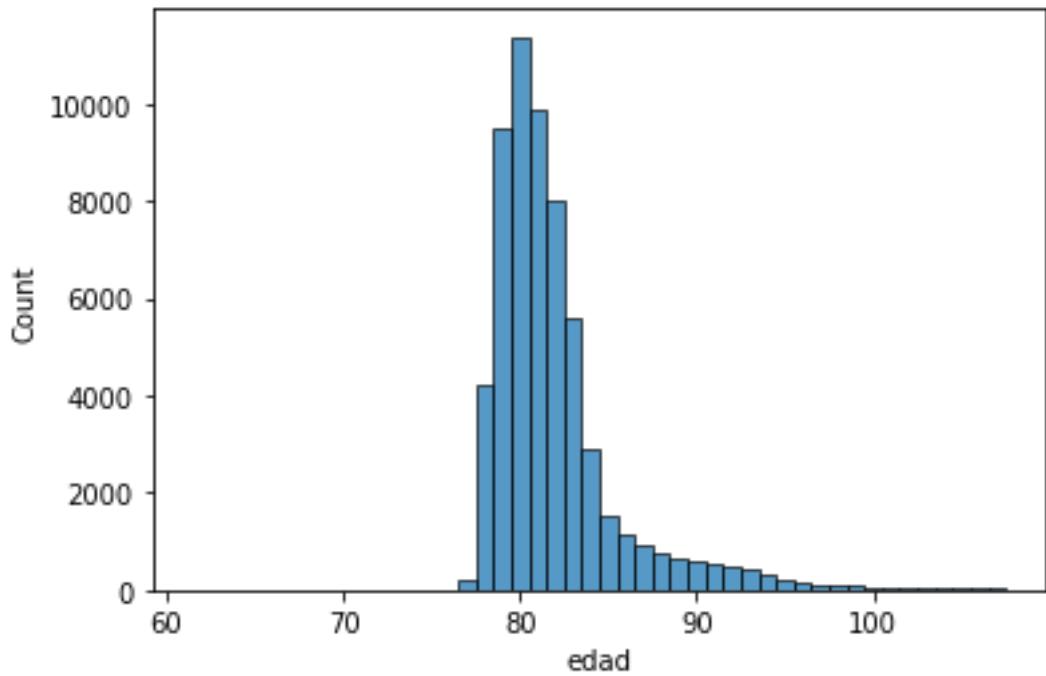
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 59999 entries, 0 to 59998
Data columns (total 23 columns):
 #   Column      Non-Null Count  Dtype  
 ---  --          -----          ----- 
 0   sexo         59999 non-null   int64  
 1   edad         59999 non-null   int64  
 2   imce         59999 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   madre_trabaja 59999 non-null   int64  
 10  madre_empleada 59999 non-null   int64  
 11  act_finser   59999 non-null   float64 
 12  area         59999 non-null   float64 
 13  desc         59999 non-null   float64 
 14  social        59999 non-null   float64 
 15  sk5          59999 non-null   int64  
 16  sk6          59999 non-null   int64  
 17  sk8          59999 non-null   int64  
 18  vive_madre   59999 non-null   int64  
 19  vive_padre   59999 non-null   int64  
 20  edad         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
```

```
[ ]: var = 'edad'
sns.histplot(df, x = var, discrete=True)
df.value_counts(var)
```

```
[ ]: edad
80    11373
81    9889
79    9479
82    7983
83    5565
78    4194
84    2873
85    1549
86    1147
87    901
88    784
89    674
90    591
91    537
92    459
93    405
94    337
95    230
77    203
96    153
97    111
98     95
```

```
99          80
101         59
103         58
100         57
105         42
102         42
104         41
106         32
107         26
75          9
76          6
74          4
73          4
71          2
65          1
69          1
68          1
66          1
62          1
dtype: int64
```

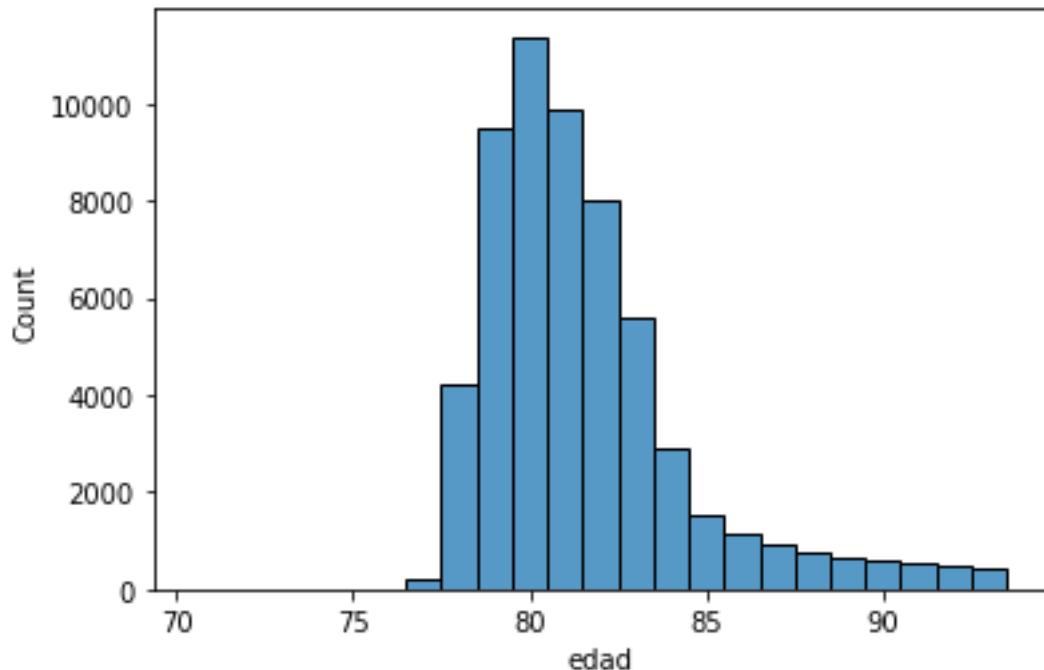


```
[ ]: df.edad.mean(), df.edad.std()
```

```
[ ]: (81.91931532192203, 3.815187887008648)
```

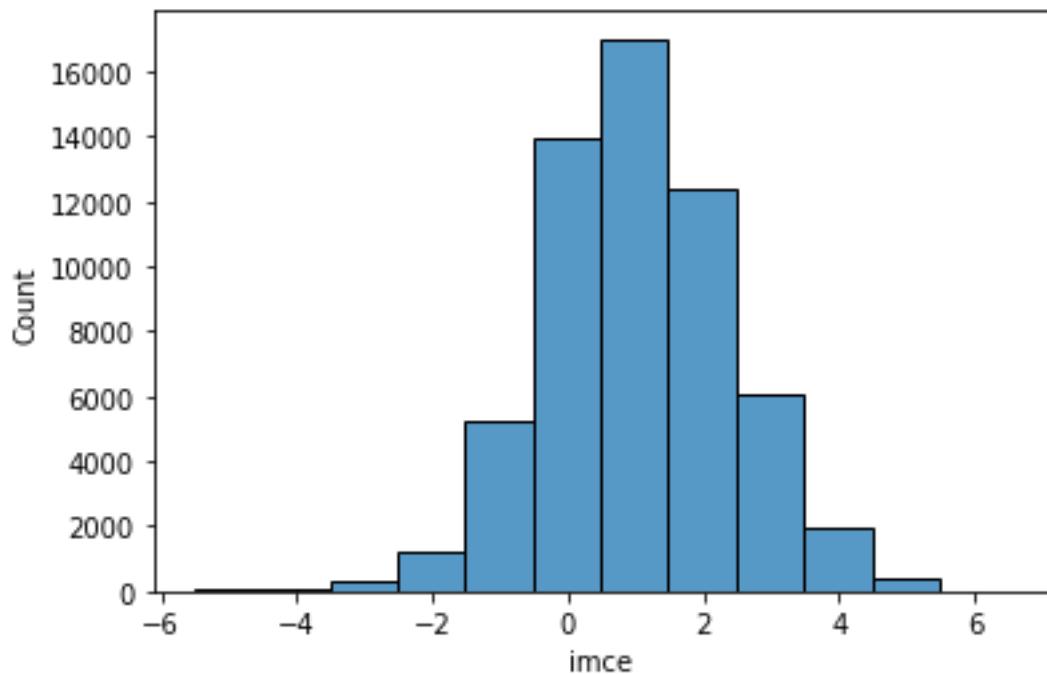
```
[ ]: df.edad.mean() - df.edad.std() * 3,df.edad.mean() + df.edad.std() * 3
[ ]: (70.47375166089608, 93.36487898294797)
[ ]: df = df[(df.edad > df.edad.mean() - df.edad.std() * 3) & (df.edad < df.edad.
    ↪mean() + df.edad.std() * 3)]
var = 'edad'
sns.histplot(df, x = var, discrete=True)
df.value_counts(var)

[ ]: edad
80      11373
81      9889
79      9479
82      7983
83      5565
78      4194
84      2873
85      1549
86      1147
87      901
88      784
89      674
90      591
91      537
92      459
93      405
77      203
75       9
76       6
73       4
74       4
71       2
dtype: int64
```



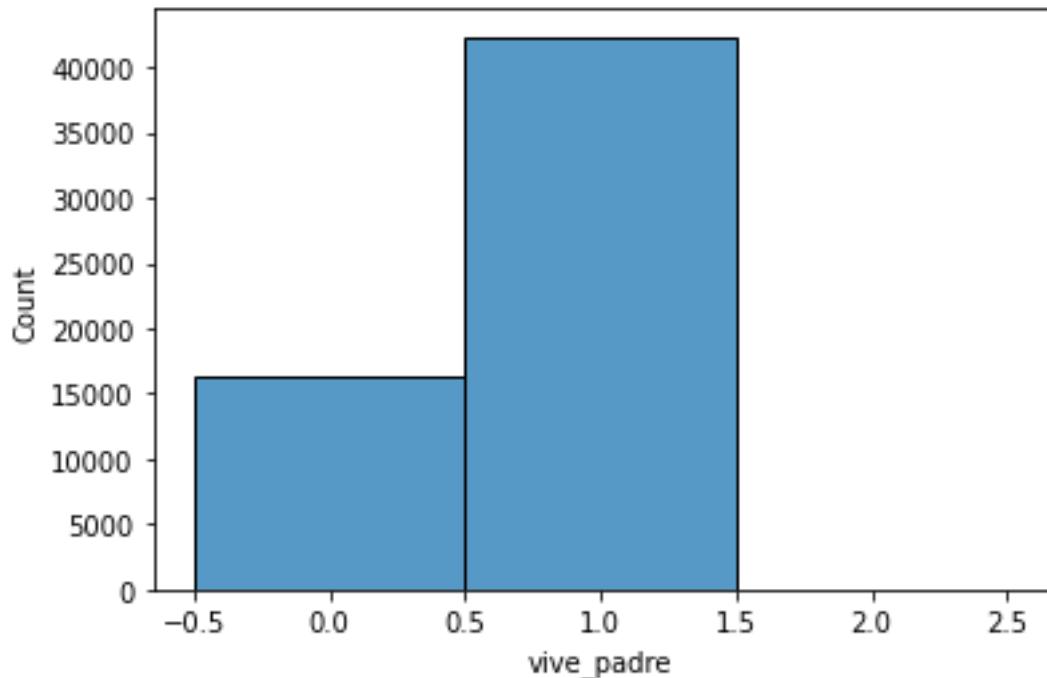
```
[ ]: var = 'imce'
sns.histplot(df, x = var, discrete=True)
df.value_counts(var)
```

```
[ ]: imce
0.74    213
0.73    206
0.39    203
0.87    203
1.07    198
...
-3.77    1
-3.75    1
-3.73    1
-3.67    1
-5.02    1
Length: 932, dtype: int64
```



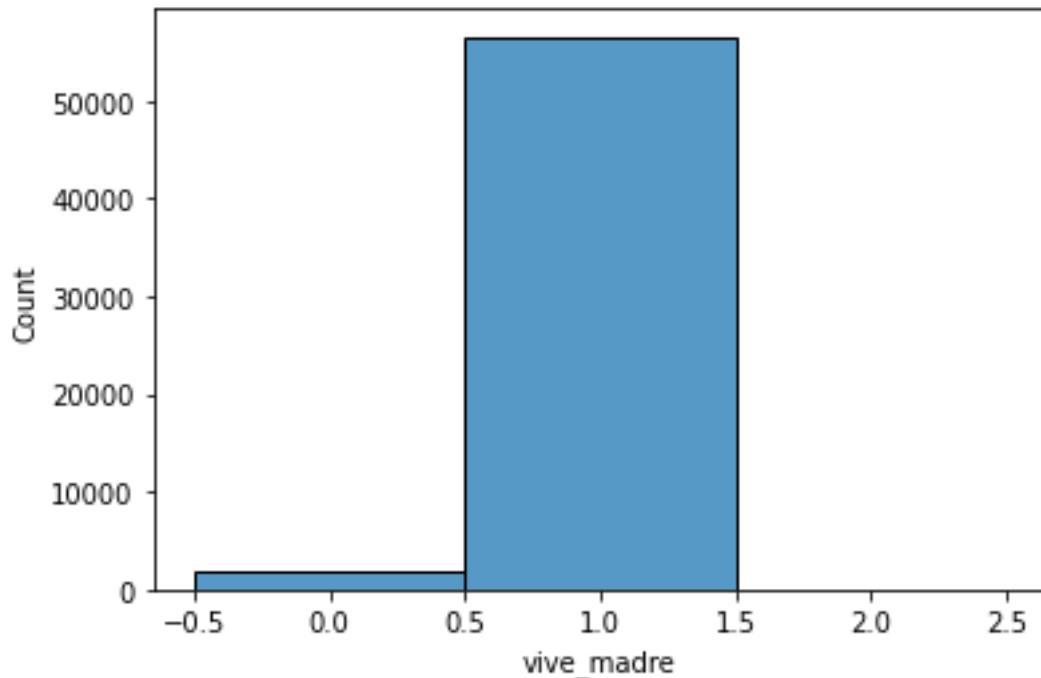
```
[ ]: var = 'vive_padre'
sns.histplot(df, x = var, discrete=True)
df.value_counts(var)
```

```
[ ]: vive_padre
1    42323
0    16285
2     23
dtype: int64
```



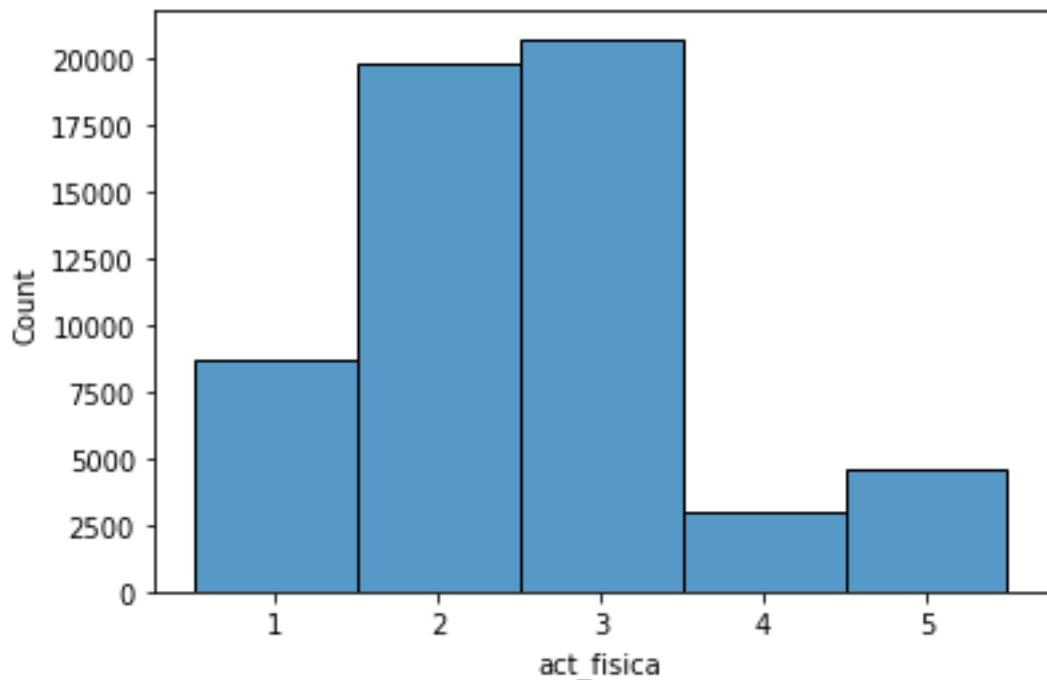
```
[ ]: var = 'vive_madre'  
sns.histplot(df, x = var, discrete=True)  
df.value_counts(var)
```

```
[ ]: vive_madre  
1    56563  
0    1980  
2     88  
dtype: int64
```



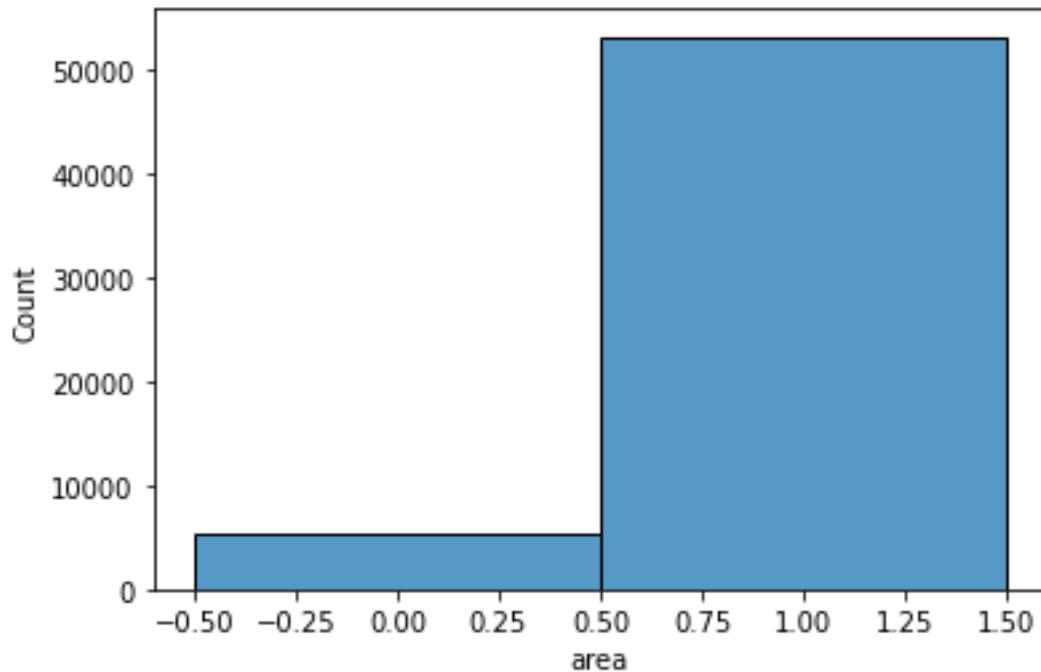
```
[ ]: var = 'act_fisica'  
sns.histplot(df, x = var, discrete=True)  
df.value_counts(var)
```

```
[ ]: act_fisica  
3.0    20725  
2.0    19779  
1.0    8662  
5.0    4621  
4.0    2957  
dtype: int64
```



```
[ ]: var = 'area'
sns.histplot(df, x = var, discrete=True)
df.value_counts(var)
```

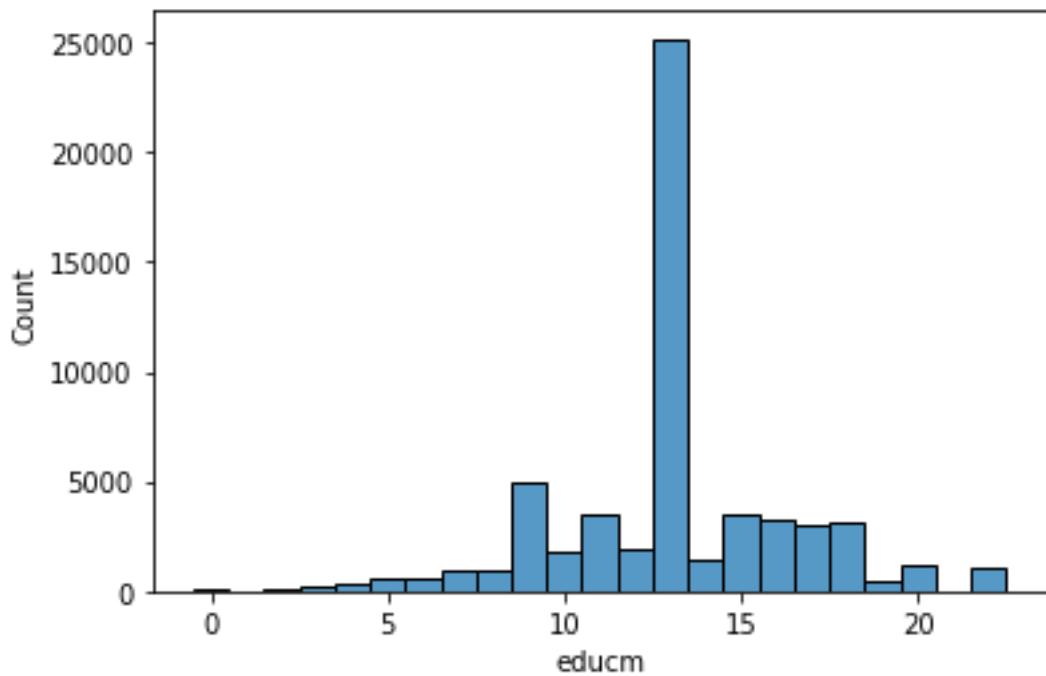
```
[ ]: area
1    53180
0    5451
dtype: int64
```



```
[ ]: var = 'educm'
sns.histplot(df, x = var, discrete=True)
df.value_counts(var)
```

```
[ ]: educm
13.0    25168
9.0     4921
15.0    3525
11.0    3462
16.0    3213
18.0    3136
17.0    3031
12.0    1908
10.0    1823
14.0    1420
20.0    1142
22.0    1018
7.0     968
8.0     962
6.0     575
5.0     532
19.0    421
4.0     314
3.0     196
0.0     113
```

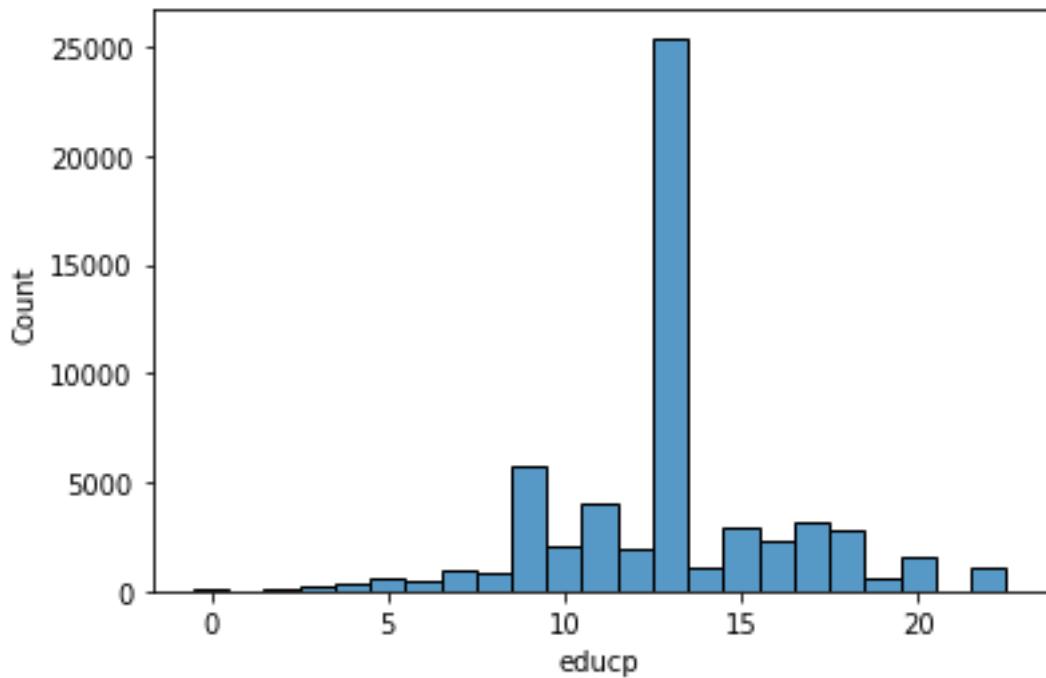
```
2.0      103  
dtype: int64
```



```
[ ]: var = 'educp'  
sns.histplot(df, x = var, discrete=True)  
df.value_counts(var)
```

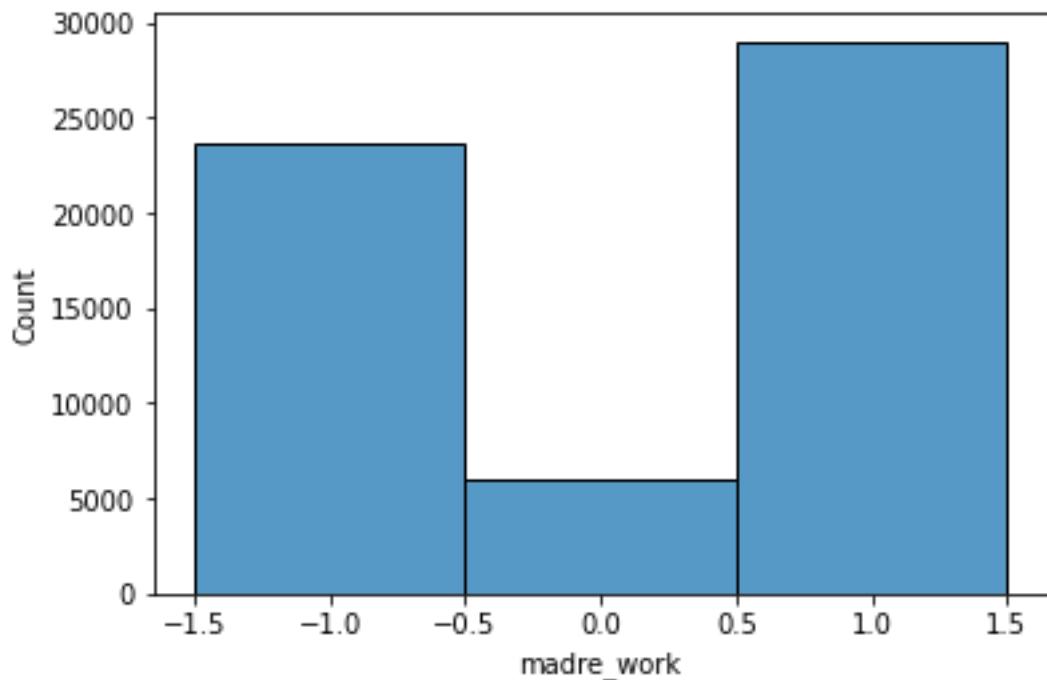
```
[ ]: educp  
13    25430  
9     5798  
11    4051  
17    3156  
15    2899  
18    2851  
16    2323  
10    2065  
12    1986  
20    1597  
22    1082  
14    1050  
7     938  
8     820  
5     620  
19    607  
6     520
```

```
4      326
3      248
0      150
2      114
dtype: int64
```



```
[ ]: # madre_work: si la madre trabaja (-1: labor domestica, 0: desempleada, 1: empleada)
var = 'madre_work'
sns.histplot(df, x = var, discrete=True)
df.value_counts(var)
```

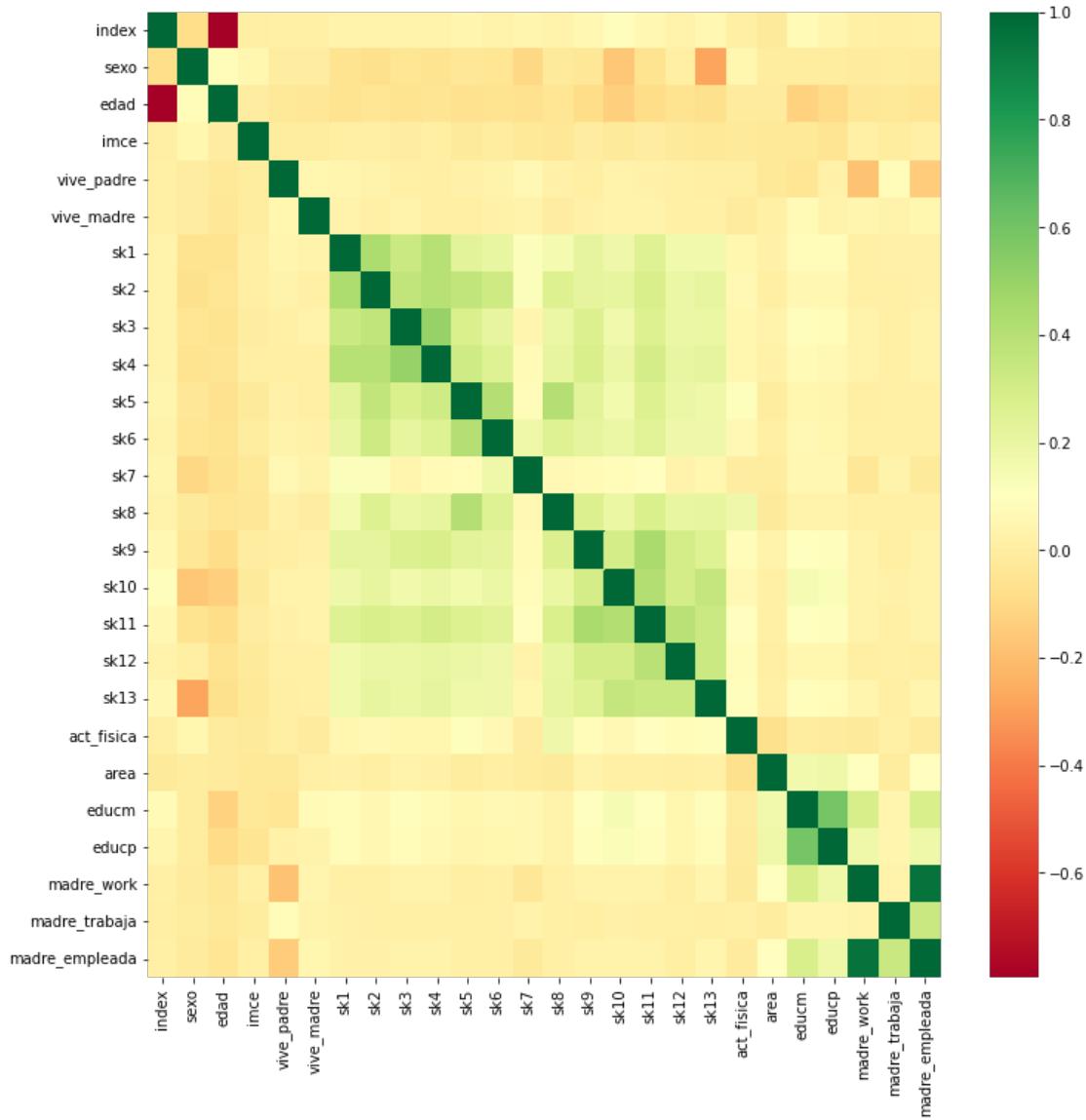
```
[ ]: madre_work
1      28996
-1     23599
0      6036
dtype: int64
```



8 Anexo 2: Heatmap

```
[ ]: fig, ax = plt.subplots(1,1, figsize = (12,12))
sns.heatmap(junaeb.corr(), cmap='RdYlGn', ax = ax)
```

```
[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbee0ae5a90>
```



En general segun lo visto en el heatmap, la correlacion entre variables es mas bien baja llegando en algunos casos 0.5, pero no superior a eso, aunque de todas formas se tomará esta informacióen en cuenta en caso de ser necesario. Importante notar que todas las correlaciones son positivas.