Tecnológico de Monterrey

Maestría en Inteligencia Artificial Aplicada

Materia | Proyecto Integrador

Laura Elena Hernández Mata | A01169213

Evelyn Aylin Rendon Medina | A01748750

Samara García González | A01273001

# **Proyecto | Islas de Calor y Justicia Social**

Avance 2 | Ingeniería de Características

Asesor: Prof. Dr. Roberto Ponce López

Tutora: María de la Paz Rico Fernández

6 de octubre de 2024

# Índice

## 1. Procesamiento de datos

A. Aplicamos operaciones comunes para convertir los datos crudos en un conjunto de variables útiles para el aprendizaje automático.

- **Generación de nuevas características**

Para la ingeniería de características, consideramos necesario analizar variables específicas en el conjunto original de datos, contemplando cuatro diferentes particiones, donde las tres primeras nos ayudan a identificar factores de vulnerabilidad y la última, atenuantes. A continuación explicaremos con mayor detalle el proceso que seguimos

**Creación de nuevas características**

Creación de característica de vulnerabilidad: "Población de 3 años y más que habla alguna lengua indígena y no habla español" o "Población en hogares censales indígenas"

In [157…

```
df_cat = df_concat_3estados.copy()

# Eliminamos los caracteres de tipo *
df_cat.replace('*', np.nan, inplace=True)
df_cat.head()
df_cat.dropna(how='all', inplace = True)

# Hacemos el cambio para que todas las variables sean numéricas
df_cat= df_cat.apply(pd.to_numeric, errors='coerce').fillna(0).astype(int)
df_cat.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 253243 entries, 0 to 253243
Columns: 102 entries, ENTIDAD to VPH_SINTIC
dtypes: int64(102)
memory usage: 199.0 MB
```

**Imagen 1.** Preparación del dataframe **.** *(Elaboración propia, 2024)*

En esta primera generación de nuevas características partimos de la concatenación de los tres Estados de interés, su respectiva limpieza e imputación designada en la etapa anterior

In [158…

```
df_indigena = df_cat.copy()
for index, row in df_indigena.iterrows():
    if row['P3HLINHE'] != 0 or row['PHOG_IND'] != 0:
        df_indigena.at[index, 'Vuln_Indigena'] = (int(df_indigena.at[index,'P3HLINHE']) + int(df_indigena.at[index,'PHOG_

print(df_indigena[['Vuln_Indigena', 'P3HLINHE', 'PHOG_IND']].head())
```

| | Vuln_Indigena | P3HLINHE | PHOG_IND |
|---|---|---|---|
| 0 | 613190.0 | 28497 | 584693 |
| 1 | 203.0 | 0 | 203 |
| 2 | 12.0 | 0 | 12 |
| 3 | 12.0 | 0 | 12 |
| 4 | NaN | 0 | 0 |

**Imagen 2.** Extracción y generación de características para población indigena**.** *(Elaboración propia, 2024)*

En este primer listado, tomamos las columnas:

- **P3HLINHE:** Población de 3 años y más que habla alguna lengua indígena y no habla español

- **PHOG_IND:** Población en hogares censales indígenas

Donde contemplamos que dicha población, para cada registro en el set de datos, tendría cierto factor de vulnerabilidad adicional al resto del total, que sería especialmente importante considerar. Así pues, si al menos alguno de los dos (condición lógica OR) registros contienen datos diferentes a cero, se sumarán; obteniendo el total de ambos como un valor del tipo entero.

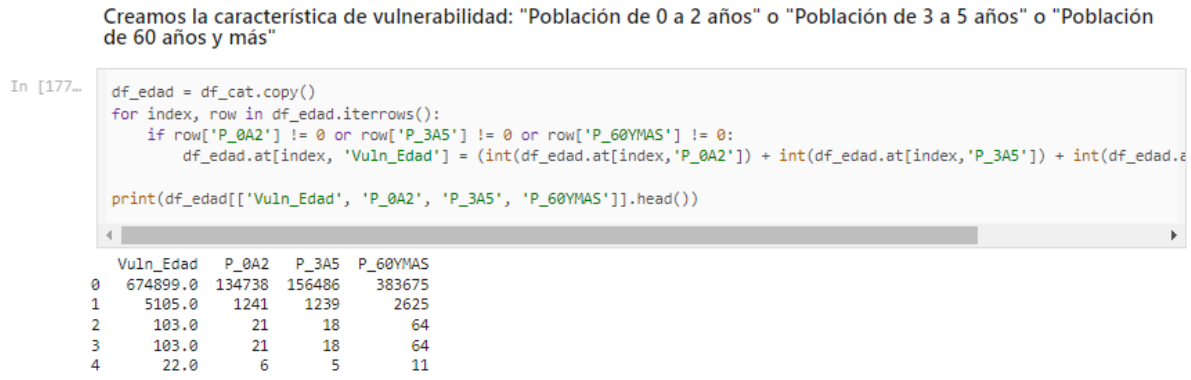


**Imagen 3.** Extracción y generación de características para de cierta edad**.** *(Elaboración propia, 2024)*

Contemplando las columnas:

- **P_0A2:** Población de 0 a 2 años

- **P_3A5:** Población de 3 a 5 años

- **P_60YMAS:** Población de 60 años y más

Seguimos una lógica parecida al caso anterior, donde identificamos cierto rango de edades que podrían posicionar a esa proporción de la población total como

vulnerable sobre el resto. Por tanto, si hay valores en al menos alguna de las tres columnas, se sumarán las personas que yacen en dicha condición, en su defecto, el valor será cero.

Creamos la característica de vulnerabilidad: "Población con discapacidad" o "Población con limitación"

In [179…

```
df_disc_lim = df_cat.copy()
for index, row in df_disc_lim.iterrows():
    if row['PCON_DISC'] != 0 or row['PCON_LIMI'] != 0:
        df_disc_lim.at[index, 'Vuln_Disc_Lim'] = (int(df_disc_lim.at[index,'PCON_DISC']) + int(df_disc_lim.at[index,'PCON_

print(df_disc_lim[['Vuln_Disc_Lim', 'PCON_DISC', 'PCON_LIMI']].head())
```

```
   Vuln_Disc_Lim  PCON_DISC  PCON_LIMI
0       580203.0     166965     413238
1         3545.0        974       2571
2          139.0         43         96
3          139.0         43         96
4           30.0          9         21
```

**Imagen 4.** Extracción y generación de características para limitaciones**.** *(Elaboración propia, 2024)*

Donde definimos como relevantes las columnas:

- **PCON_DISC:** Población con discapacidad

- **PCON_LIMI:** Población con limitación

Por tanto, contemplando un proceder similar al anteriormente expuesto, seccionamos las personas que poseen alguna discapacidad en general o también cierta limitación. Se abarca, por tanto, el aspecto físico y cognitivo que, sin duda podría resultar en un mayor índice respecto a la vulnerabilidad

Creamos la característica de vulnerabilidad: "Población de 15 años y más analfabeta" o "Población de 15 años y más sin escolaridad" o "Población sin afiliación a servicios de salud"

In [186…

```
df_esc_salud = df_cat.copy()
for index, row in df_esc_salud.iterrows():
    if row['P15YM_AN'] != 0 or row['P15YM_SE'] != 0 or row['PSINDER']:
        df_esc_salud.at[index, 'Vuln_Esc_Salud'] = (int(df_esc_salud.at[index,'P15YM_AN']) + int(df_esc_salud.at[index,'P1

print(df_esc_salud[['Vuln_Esc_Salud', 'P15YM_AN', 'P15YM_SE', 'PSINDER']].head())
```

```
   Vuln_Esc_Salud  P15YM_AN  P15YM_SE  PSINDER
0       1222960.0    151311    143099   928550
1         13447.0      1755      1894     9798
2           251.0        10        14      227
3           251.0        10        14      227
4            50.0         0         3       47
```

**Imagen 5.** Extracción y generación de características para la escolaridad**.** *(Elaboración propia, 2024)*

En este punto consideramos las columnas:

- **P15YM_AN:** Población de 15 años y más analfabeta

- **P15YM_SE:** Población de 15 años y más sin escolaridad

- **PSINDER:** Población sin afiliación a servicios de salud

Dada la condición y procesamiento lógico ya manifestado, realizamos lo mismo para esta partición de la población total, considerando la vulnerabilidad relativa a falta de educación y servicios de salud

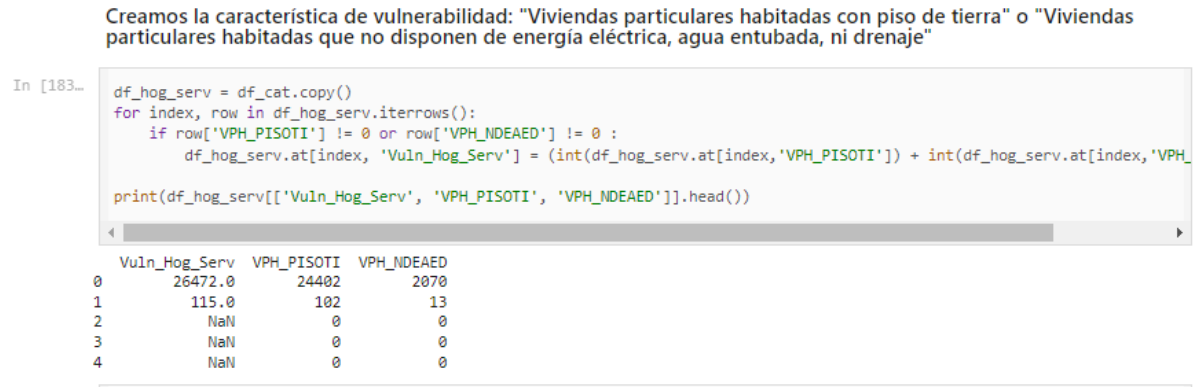


**Imagen 6.** Extracción y generación de características para la vivienda**.** *(Elaboración propia, 2024)*

En términos de vivienda, consideramos en esta primera partición las columnas:

- **VPH_PISOTI:** Viviendas particulares habitadas con piso de tierra

- **VPH_NDEAED:** Viviendas particulares habitadas que no disponen de energía eléctrica, agua entubada, ni drenaje

Se destacan estas variables porque nos ayudan a entender las condiciones de la vivienda para las personas en cuestión. Además, nos ayuda a revelar cierto indicio de su situación socioeconómica, donde, de pertenecer a dicha partición, podría considerar como un factor adicional de vulnerabilidad

Creamos la característica de vulnerabilidad: "Viviendas particulares habitadas sin radio ni televisor" o "Viviendas particulares habitadas sin línea telefónica fija ni teléfono celular" o "Viviendas particulares habitadas sin computadora ni Internet"

In [188…

```
df_hog_connect = df_cat.copy()
for index, row in df_hog_connect.iterrows():
    if row['VPH_SINRTV'] != 0 or row['VPH_SINLTC'] != 0 or row['VPH_SINCINT']:
        df_hog_connect.at[index, 'Vuln_Hog_Connect'] = (int(df_hog_connect.at[index,'VPH_SINRTV']) + int(df_hog_connect.at

print(df_hog_connect[['Vuln_Hog_Connect', 'VPH_SINRTV', 'VPH_SINLTC', 'VPH_SINCINT']].head())
```

| | Vuln_Hog_Connect | VPH_SINRTV | VPH_SINLTC | VPH_SINCINT |
|---|---|---|---|---|
| 0 | 621072.0 | 51691 | 102005 | 467376 |
| 1 | 4707.0 | 322 | 680 | 3705 |
| 2 | 77.0 | 9 | 8 | 60 |
| 3 | 77.0 | 9 | 8 | 60 |
| 4 | 21.0 | 4 | 3 | 14 |

**Imagen 7.** Extracción y generación de características para los bienes**.** *(Elaboración propia, 2024)*

Para este punto, respecto a lo que se puede poseer en la vivienda, consideramos:

- **VPH_SINRTV:** Viviendas particulares habitadas sin radio ni televisor

- **VPH_SINLTC:** Viviendas particulares habitadas sin línea telefónica fija ni teléfono celular

- **VPH_SINCINT:** Viviendas particulares habitadas sin computadora ni Internet

En este punto nos referimos más a la carencia de bienes, los cuales si bien podrían resultar un atenuante en el grado de vulnerabilidad, al no disponerlos, resultaría en un efecto contrario

Creamos la característica atenuante: "Viviendas particulares habitadas que disponen de refrigerador" o "Viviendas particulares habitadas que disponen de automóvil o camioneta" o "Viviendas particulares habitadas que disponen de motocicleta o motoneta" o "Viviendas particulares habitadas que disponen de bicicleta como medio de transporte"

In [192…

```
df_hog_atenuante = df_cat.copy()
for index, row in df_hog_atenuante.iterrows():
    if row['VPH_REFRI'] != 0 or row['VPH_AUTOM'] != 0 or row['VPH_MOTO'] != 0 or row['VPH_BICI']:
        df_hog_atenuante.at[index, 'Vuln_Hog_Atenuante'] = (int(df_hog_atenuante.at[index,'VPH_REFRI']) + int(df_hog_atenu

print(df_hog_atenuante[['Vuln_Hog_Atenuante', 'VPH_REFRI', 'VPH_AUTOM', 'VPH_MOTO', 'VPH_BICI']].head())
```

| | Vuln_Hog_Atenuante | VPH_REFRI | VPH_AUTOM | VPH_MOTO | VPH_BICI |
|---|---|---|---|---|---|
| 0 | 1365712.0 | 708258 | 378990 | 86059 | 192405 |
| 1 | 12162.0 | 4986 | 3572 | 1273 | 2331 |
| 2 | 238.0 | 114 | 71 | 10 | 43 |
| 3 | 238.0 | 114 | 71 | 10 | 43 |
| 4 | 33.0 | 22 | 8 | 0 | 3 |

**Imagen 8.** Extracción y generación de características para los atenuantes**.** *(Elaboración propia, 2024)*

Finalmente, para esta última característica atenuante en cuanto a la vulnerabilidad, consideramos:

- **VPH_REFRI:** Viviendas particulares habitadas que disponen de refrigerador

- **VPH_AUTOM:** Viviendas particulares habitadas que disponen de automóvil o camioneta

- **VPH_MOTO:** Viviendas particulares habitadas que disponen de motocicleta o motoneta

- **VPH_BICI:** Viviendas particulares habitadas que disponen de bicicleta como medio de transporte

En esta última sumarización se entienden los factores que podrían atenuar una situación vulnerable, como también revelar más datos sobre una posición socioeconómica más saludable.

El procesamiento y generación de grupos anteriormente expuestos fue de relevancia para poder particionar del total poblacional, aquellos que se podrían ver especialmente vulnerables, generando un totalizador para lo designado. Dicho resumen nos permitirá agrupar y cuantificar las personas que designamos vulnerables, lo que se detalla más adelante.

| | Vuln_Indigena | Vuln_Edad | Vuln_Disc_Lim | Vuln_Esc_Salud | Vuln_Hog_Serv | Vuln_Hog_Connect | Vuln_Hog_Atenuante |
|---|---|---|---|---|---|---|---|
| count | 2.532430e+05 | 2.532430e+05 | 2.532430e+05 | 2.532430e+05 | 253243.000000 | 2.532430e+05 | 2.532430e+05 |
| mean | 2.997133e+01 | 1.134401e+02 | 8.965603e+01 | 1.893087e+02 | 1.988793 | 5.853404e+01 | 2.520060e+02 |
| std | 2.529480e+03 | 8.533440e+03 | 6.654507e+03 | 1.480060e+04 | 206.042487 | 4.919776e+03 | 1.854559e+04 |
| min | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000e+00 | 0.000000 | 0.000000e+00 | 0.000000e+00 |
| 25% | 0.000000e+00 | 4.000000e+00 | 0.000000e+00 | 7.000000e+00 | 0.000000 | 0.000000e+00 | 1.200000e+01 |
| 50% | 0.000000e+00 | 1.400000e+01 | 1.000000e+01 | 2.200000e+01 | 0.000000 | 5.000000e+00 | 3.200000e+01 |
| 75% | 5.000000e+00 | 3.200000e+01 | 2.500000e+01 | 5.100000e+01 | 0.000000 | 1.400000e+01 | 6.700000e+01 |
| max | 1.031962e+06 | 3.478198e+06 | 2.685381e+06 | 6.497550e+06 | 96753.000000 | 2.232460e+06 | 7.509537e+06 |

**Imagen 9.** Medidas de distribución de las variables generadas**.** *(Elaboración propia, 2024)*

Consecuentemente, después de añadir o concatenar el total de población vulnerable y el atenuante considerado en el set de datos ya procesado, obtenemos las medidas

de distribución central para entender mejor el comportamiento de los datos y cuántas personas podrían considerarse vulnerables, que es el enfoque que se tiene en el proyecto integrador, respecto a la justicia social y establecer un nivel de criticidad en las acciones.

- **Discretización o *binning***

Agregamos una columna adicional a nuestro resultado anterior para representar los rangos discretizados por rangos como valores enteros. Esto nos permitirá realizar análisis cuantitativos más fácilmente. En este caso realizamos la discretización en 3 grupos dependiendo de los valores obtenidos en 'Vuln_Esc_Salud' , utilizando los siguientes bines = [0, 1000, 15000, np.inf] siendo:

- **Alto:** Valores en 'Vuln_Esc_Salud' de 0 a 1000, sin contemplar 1000

- **Medio:** Valores en 'Vuln_Esc_Salud' de 0 a 15000, sin contemplar 15000

- **Bajo:** Valores en 'Vuln_Esc_Salud' de 15000 al infinito

```
# Bajo 0-1000, Medio 1000-15000, Alto 15000<
bins = [0, 1000, 15000, np.inf]
labels = ['Bajo', 'Medio', 'Alto']

# Añadiendo  las columnas
df_esc_salud['Vuln_Salud_Discreta'] = pd.cut(df_esc_salud['Vuln_Esc_Salud'], bins=bins, labels=labels)
df_esc_salud['Vuln_Salud_Discreta_Int'] = df_esc_salud['Vuln_Salud_Discreta'].cat.codes + 1

print(df_esc_salud[['Vuln_Esc_Salud', 'Vuln_Salud_Discreta','Vuln_Salud_Discreta_Int']].head())
```

**Imagen 10.** Discretización de variables. *(Elaboración propia, 2024)*

Además se colocó una columna extra en la que se muestra el grupo al que pertenece pero en forma de número entero para facilitar su uso en futuras operaciones.

| | Vuln_Esc_Salud | Vuln_Salud_Discreta | Vuln_Salud_Discreta_Int |
|---|---|---|---|
| 0 | 1222960.0 | Alto | 3 |
| 1 | 13447.0 | Medio | 2 |
| 2 | 251.0 | Bajo | 1 |
| 3 | 251.0 | Bajo | 1 |
| 4 | 50.0 | Bajo | 1 |

**Imagen 11.** Resultados de discretización. *(Elaboración propia, 2024)*

- **Codificación**

De forma ilustrativa, se genera una codificación del tipo *Label Encoding*, para el nombre de la entidad, que era nuestra única variable categórica. Sin embargo, solamente se hace para seguir la lógica del proceso, lo que sería necesario de no contar con la columna “ENTIDAD”, que ya contiene de forma numérica un equivalente para los nombres de la entidad. A continuación, se muestra la codificación realizada en nuestro DataFrame.

De forma ilustrativa, codificamos la variable "NOM_ENT"

In [155…

```
df_concat_3copy = df_concat_3estados.copy()
label_encoder = LabelEncoder()
df_concat_3copy['NOM_ENT'] = label_encoder.fit_transform(df_concat_3copy['NOM_ENT'])
df_concat_3copy.head()
```

Out[155…

| | ENTIDAD | NOM_ENT | LOC | AGEB | MZA | POBTOT | POBFEM | POBMAS | P_0A2 | P_0A2_F | ... | VPH_RADIO | VPH_TV | VPH_PC | VPI |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 13.0 | 1 | 0.0 | 0000 | 0.0 | 3082841.0 | 1601462 | 1481379 | 134738 | 66770 | ... | 579898 | 759456 | 261093 | |
| 1 | 13.0 | 1 | 0.0 | 0000 | 0.0 | 22268.0 | 11563 | 10705 | 1241 | 593 | ... | 3835 | 5303 | 1045 | |
| 2 | 13.0 | 1 | 1.0 | 0000 | 0.0 | 439.0 | 229 | 210 | 21 | 6 | ... | 76 | 116 | 39 | |
| 3 | 13.0 | 1 | 1.0 | 0043 | 0.0 | 439.0 | 229 | 210 | 21 | 6 | ... | 76 | 116 | 39 | |
| 4 | 13.0 | 1 | 1.0 | 0043 | 1.0 | 92.0 | 54 | 38 | 6 | * | ... | 12 | 21 | 7 | |

5 rows × 102 columns

**Imagen 12.** Label encoding para nombre de la entidad. *(Elaboración propia, 2024)*

Asimismo, observamos los primeros registros obtenidos y que ya están codificados, sin embargo, dicho cálculo será descartado posteriormente por lo ya comentado. En este punto cabe destacar que seleccionamos la codificación conocida como *Label Encoding* porque no aumenta la dimensionalidad del set de datos, contrario a *One Hot Encoder*. Lo anterior es relevante dada la cantidad de registros y columnas de nuestro conjunto.

- **Escalamiento**

A continuación, se muestra el escalamiento realizado al DataFrame; esto a través de StandardScaler.

```
In [ ]: scaler = StandardScaler()
        df_transformed = pd.DataFrame(scaler.fit_transform(df2), columns=df2.columns)
        df_transformed.head()
```

**Imagen 13.** Método para realizar escalamiento. *(Elaboración propia, 2024)*

Es importante realizar escalamiento o al menos tener un abordaje y verificar que los resultados sean los esperados puesto que, posteriormente dado el modelo de clusterización seleccionado, podría haber cierta sensibilidad a valores que no estén normalizados, afectando el rendimiento general del mismo.

Asimismo, normalizar puede ser relevante en datos poblacionales para entender mejor el conjunto de datos, en perspectiva de comparar las distintas densidades poblacionales y que se mantenga una proporción que pueda ser representativa.

* Todas las decisiones y técnicas empleadas deben ser justificadas.

## 2. Métodos de filtrado

B. Se utilizaron métodos de filtrado para la selección de características y técnicas de extracción de características. Esto nos permitirá reducir los requerimientos de almacenamiento, la complejidad del modelo y el tiempo de entrenamiento.

- **Correlación**

En términos de correlación en nuestras variables, lógicamente encontramos valores altos, sobre todo hablando de una correlación positiva, tal como se muestra en la siguiente imagen. Cabe destacar que consideramos el comportamiento observado como racional puesto que, básicamente el set de datos contiene variables relativas a la población, destacando un totalizador y particiones de la misma.

Recordemos que aquellas variables que tienen una correlación perfecta son las que tienen un 1; lo que ocurrirá siempre que se analiza una variable consigo misma, siendo lo que observamos en la diagonal de la imagen. Sin embargo, se puede apreciar que hay, por ejemplo, una alta correlación cuando hablamos de personas de 60 años o más con alguna discapacidad.

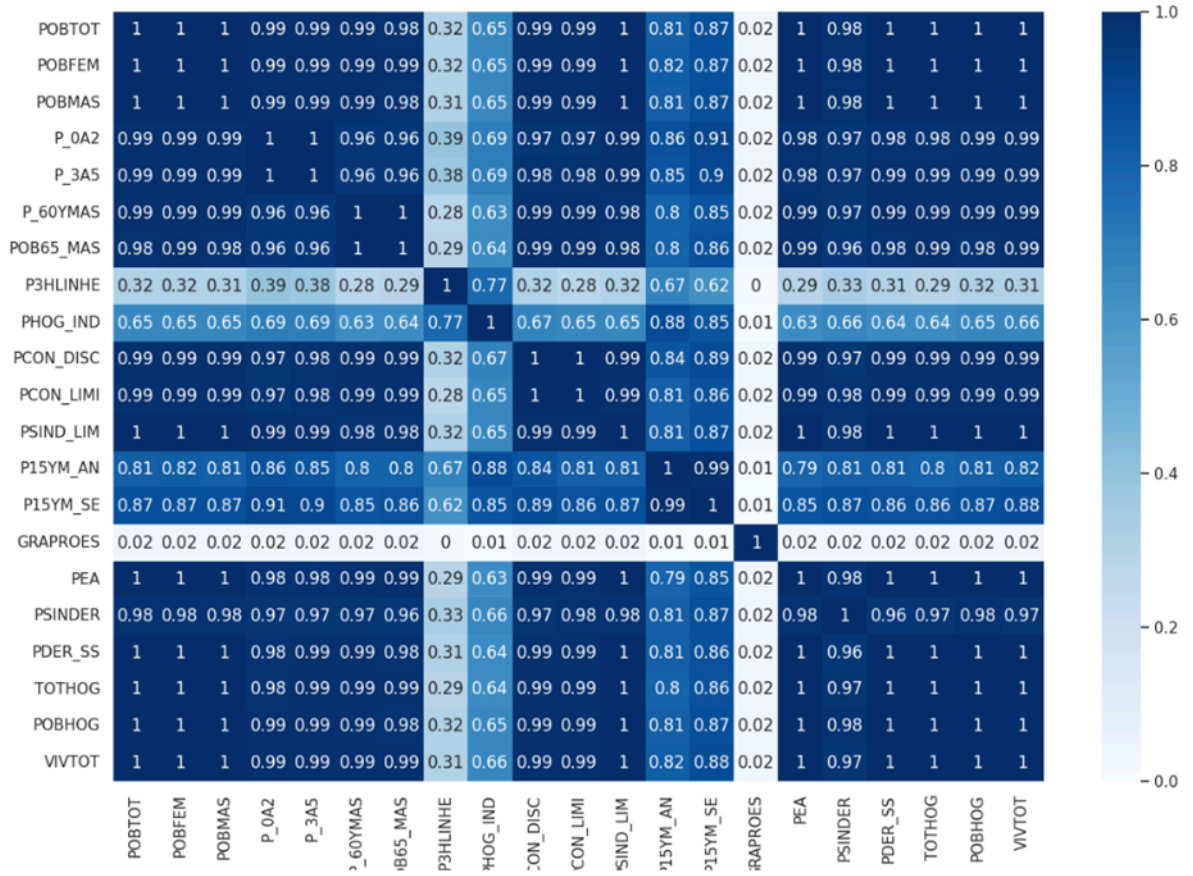


**Imagen 14.** Mapa de correlación **.** *(Elaboración propia, 2024)*

El mismo comportamiento lo podemos observar si calculamos e imprimimos en forma de texto el cálculo de la correlación entre las variables, que es lo que observamos a continuación.

**Correlación**

In [96]:
```
print(df2.corr())
```

```
              POBTOT    POBFEM    POBMAS     P_0A2     P_3A5  P_60YMAS  \
POBTOT      1.000000  0.999980  0.999977  0.988880  0.991567  0.985677
POBFEM      0.999980  1.000000  0.999914  0.988007  0.990801  0.986654
POBMAS      0.999977  0.999914  1.000000  0.989768  0.992340  0.984590
P_0A2       0.988880  0.988007  0.989768  1.000000  0.999762  0.950209
P_3A5       0.991567  0.990801  0.992340  0.999762  1.000000  0.956023
P_60YMAS    0.985677  0.986654  0.984590  0.950209  0.956023  1.000000
POB65_MAS   0.981622  0.982737  0.980388  0.943158  0.949355  0.999710
P3HLINHE    0.314569  0.315068  0.314022  0.336454  0.335232  0.298974
PHOG_IND    0.908959  0.908071  0.909866  0.935688  0.933555  0.861122
PCON_DISC   0.995731  0.996195  0.995191  0.972942  0.977191  0.995252
PCON_LIMI   0.998698  0.998913  0.998424  0.982272  0.985751  0.990796
PSIND_LIM   0.999914  0.999827  0.999963  0.990384  0.992846  0.983713
P15YM_AN    0.952003  0.950954  0.953082  0.976324  0.974511  0.903514
P15YM_SE    0.978583  0.977759  0.979419  0.991928  0.991533  0.939928
GRAPROES    0.013019  0.013121  0.012909  0.011100  0.011432  0.014743
PEA         0.999072  0.999297  0.998788  0.981726  0.985171  0.991620
PSINDER     0.996131  0.995596  0.996657  0.996873  0.998108  0.968215
PDER_SS     0.999116  0.999343  0.998829  0.982341  0.985716  0.991345
TOTHOG      0.998757  0.999031  0.998420  0.980804  0.984347  0.992253
POBHOG      0.999998  0.999973  0.999981  0.989055  0.991722  0.985483
VIVTOT      0.998988  0.999112  0.998811  0.986299  0.989190  0.987020
```

**Imagen 15.** Correlación **.** *(Elaboración propia, 2024)*

Lo anterior nos puede permitir deducir, por ejemplo, en qué proporción aumenta cierta condición de la población contra el total, es decir, generar estadísticas de incidencias o relativos al nivel socioeconómico contra un total para cada región.

Por otro lado, para ahondar un poco más en la correlación, sabemos que un comportamiento lineal es necesario, por lo cual decidimos generar un diagrama de dispersión que nos permitiese entender la distribución de los datos contra una línea de referencia, que es lo que se observa en la imagen que se presenta a continuación.

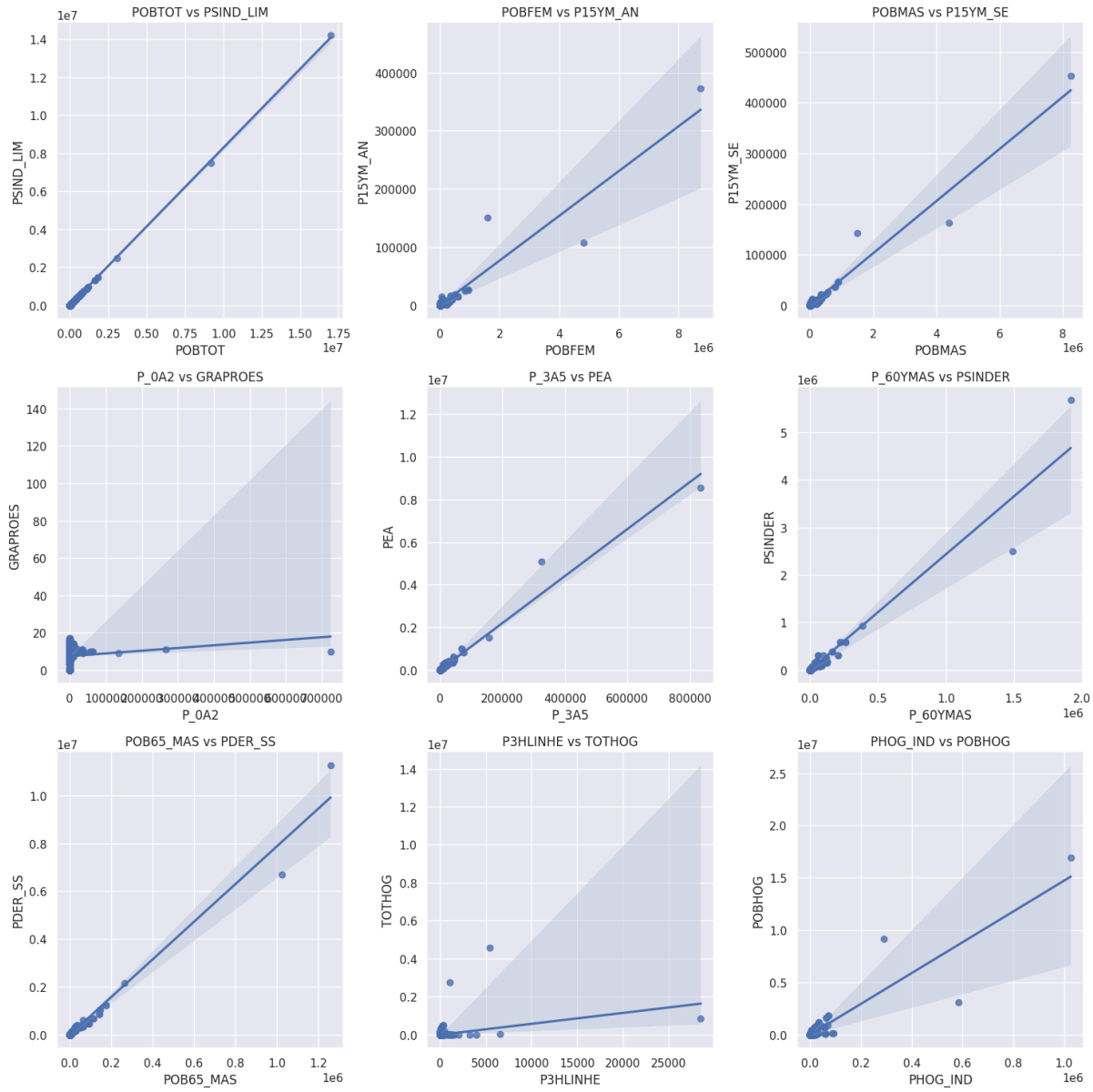


**Imagen 16.** Diagrama de dispersión con línea de referencia**.** *(Elaboración propia, 2024)*

Como se observa, los datos que contiene nuestro set, -o la concatenación de nuestros conjuntos originales-, presenta un comportamiento, en su mayoría, lineal. En algunos casos se destacan algunos datos atípicos.

- **Análisis de componentes principales (PCA)**

Como parte de esta y la anterior entrega, se realizó un análisis PCA para poder entender mejor cómo se comportan las variables entre sí y, de esta manera, conocer los escenarios a los cuales nos podremos enfrentar más adelante cuando

trabajemos con clústeres más avanzados. Así pues, en la siguiente imagen podemos apreciar el resumen del análisis PCA, así como también la proporción acumulada de cada variable en la explicación del fenómeno.

```
[117] pca = PCA()
      pca.fit(df_transformed)
      pcaSummary = pd.DataFrame({'Standard deviation': np.round(np.sqrt(pca.explained_variance_), 2),
                                 'Proportion of variance': np.round(pca.explained_variance_ratio_ * 100, 2),
                                 'Cumulative proportion': np.round(np.cumsum(pca.explained_variance_ratio_) * 100, 2)
                                 })

      pcaSummary = pcaSummary.transpose()
      pcaSummary.columns = ['PC{}'.format(i) for i in range(1, len(pcaSummary.columns) + 1)]
      pcaSummary
```

| | PC1 | PC2 | PC3 | PC4 | PC5 | PC6 | PC7 | PC8 | PC9 | PC10 | ... | PC12 | PC13 | PC14 | PC15 | PC16 | PC17 | PC18 | PC19 | PC20 | PC21 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Standard deviation | 4.24 | 1.27 | 1.00 | 0.42 | 0.36 | 0.27 | 0.21 | 0.08 | 0.07 | 0.06 | ... | 0.03 | 0.02 | 0.01 | 0.01 | 0.01 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Proportion of variance | 85.50 | 7.64 | 4.76 | 0.84 | 0.62 | 0.36 | 0.21 | 0.03 | 0.02 | 0.02 | ... | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Cumulative proportion | 85.50 | 93.13 | 97.89 | 98.73 | 99.35 | 99.70 | 99.92 | 99.94 | 99.97 | 99.98 | ... | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |

3 rows × 21 columns

**Imagen 17.** Resumen análisis PCA. *(Elaboración propia, 2024)*

Cabe destacar que para generar una mejor comprensión sobre el resultado anteriormente expresado, se grafican los componentes principales. En la imagen que se presenta a continuación es posible observar que a través de los primeros tres componentes se tendría básicamente la totalidad de la varianza explicada.

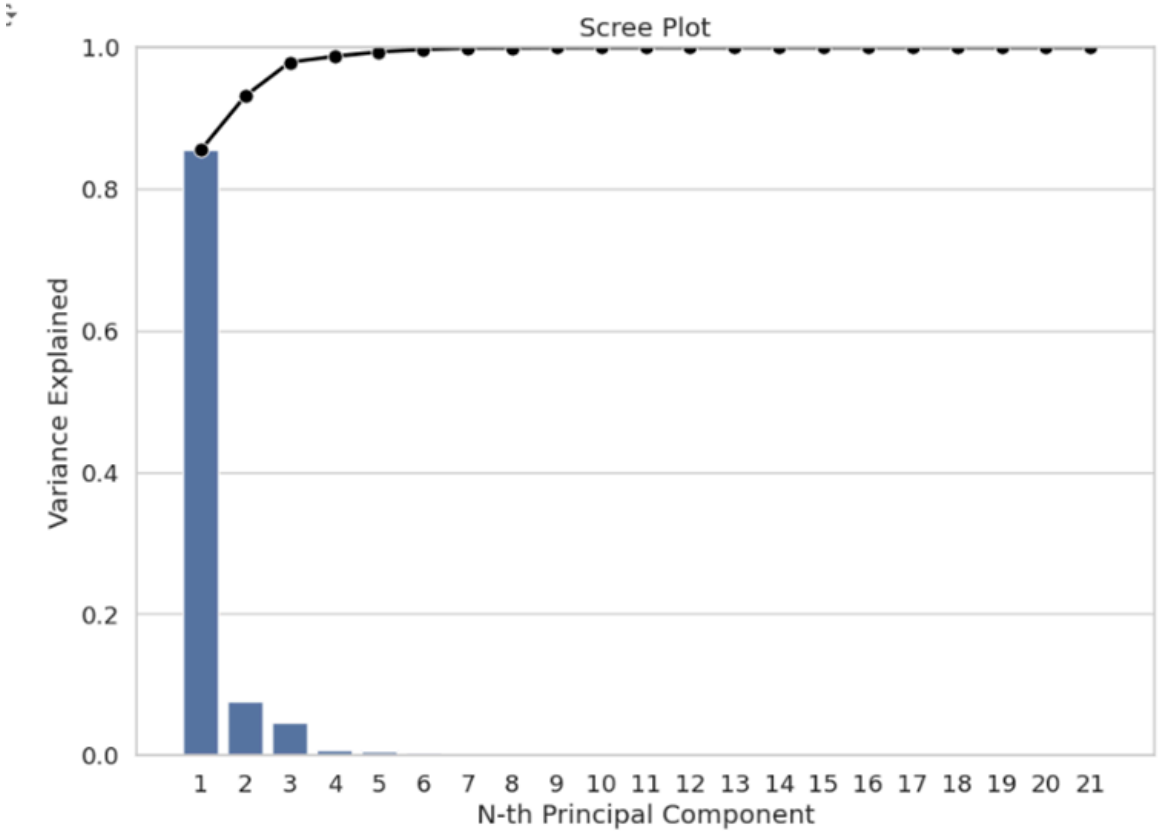

**Imagen 18.** Gráfica análisis PCA. *(Elaboración propia, 2024)*

Realizar un análisis PCA en este caso particular nos resultó relevante porque se estaban contemplando diversas variables relativas a la población y era necesario destacar aquellas que realmente generan variabilidad o resultan componentes principales para entender en su generalidad al set de datos.

- **Análisis factorial (FA)**

Una vez identificadas las principales características , decidimos aplicar un análisi factorial (AF) que es una técnica estadística multivariante que busca identificar grupos de variables altamente correlacionadas (factores) , esto debido a que al tener una gran cantidad de variables socioeconómicas en nuestro data frame, agrupar las variables altamente relacionadas nos permite simplificar la interpretación de los datos haciendo más fácil identificar patrones y tendencias.

Para esto utilizamos el resultado obtenido del paso anterior, es decir de los componentes principales (PCA), usado para crear nuevas variables (componentes principales) que son combinaciones lineales de las variables originales:

```
#Análisis Factorial
!pip install factor_analyzer
from factor_analyzer import FactorAnalyzer

corr_matrix = pcsComponents_df.corr()
print("** CORR MATRIX:" )
print(corr_matrix)
if np.isnan(corr_matrix).any().any():
    print("La matriz de correlación contiene valores NaN. Revisa tus datos.")
else:
    # Bartlett
    chi_square_value,p_value=calculate_bartlett_sphericity(corr_matrix)
    print(chi_square_value, p_value)
# Correlación entre las variables- Barlett - chi-square
from factor_analyzer.factor_analyzer import calculate_bartlett_sphericity
chi_square_value,p_value=calculate_bartlett_sphericity(corr_matrix)
print("** Barlett:")
print(chi_square_value, p_value)
# KMO
from factor_analyzer.factor_analyzer import calculate_kmo
kmo_all,kmo_model=calculate_kmo(corr_matrix)
print("** KMO:")
print(kmo_all)

fa = FactorAnalyzer(n_factors=2, rotation='varimax')
fa.fit(corr_matrix)

print("** Cargas factoriales:")
print(fa.loadings_)
print("** Varianza por cada factor:")
print(fa.get_factor_variance())
```

**Imagen 19.** Fragmento de código para análisis factorial. *(Elaboración propia, 2024)*

A continuación se muestran los resultados:

```
** Barlett:
440.202738880101 2.021664924206898e-18
** KMO:
[0.75607866 0.2515248  0.56923937 0.06373387 0.04607258 0.0320617
 0.03071883 0.03274997 0.02997418 0.02989914 0.02951193 0.02980523
 0.02933598 0.02951386 0.02934814 0.02953839 0.02943022 0.02949071
 0.02950377 0.0294569  0.02944204]
```

**Imagen 20.** Resultado Barlette y KMOl. *(Elaboración propia, 2024)*

Bartlett y KMO indican si los datos son adecuados para realizar un análisis factorial, en este caso podemos observar que obtuvimos resultados de chi-cuadrad(440.202738880101): Este valor representa el estadístico chi-cuadrado calculado. Un valor alto de chi-cuadrado sugiere que existe una diferencia significativa entre la matriz de correlación observada y una matriz identidad. En este caso, el valor es bastante alto, lo cual es una indicación de que hay una correlación significativa entre las variables.

p_value (2.021664924206898e-18): Este valor es el p-valor asociado al estadístico chi-cuadrado. Representa la probabilidad de obtener un valor de chi-cuadrado tan grande o más grande, asumiendo que la hipótesis nula es cierta. En este caso, el p-valor es extremadamente pequeño (casi cero), lo que significa que es muy poco probable obtener estos resultados por casualidad si las variables no estuvieran correlacionadas.

```
** Cargas factoriales:
[[ 1.04111706e+00 -7.62759113e-02]
 [-4.00137111e-01  2.09708414e-02]
 [-8.35481753e-01  8.16469479e-02]
 [-1.52755544e-01 -9.73855681e-03]
 [ 1.13541015e-01 -4.25311981e-02]
 [ 3.80443065e-02 -3.31842898e-02]
 [ 2.47699939e-02 -3.15482627e-02]
 [ 1.28299268e-01  9.95119500e-01]
 [ 1.46711545e-02 -3.03046497e-02]
 [ 1.34158311e-02 -3.01501215e-02]
 [ 5.32759379e-03 -2.91547548e-02]
 [ 1.17414164e-02 -2.99440226e-02]
 [-6.39673318e-04 -2.84206958e-02]
 [ 5.37925220e-03 -2.91611104e-02]
 [-9.27560093e-05 -2.84879646e-02]
 [ 6.01849855e-03 -2.92397615e-02]
 [ 2.94219637e-03 -2.88612881e-02]
 [ 4.74845011e-03 -2.90835011e-02]
 [ 5.10779267e-03 -2.91277118e-02]
 [ 3.77111735e-03 -2.89632629e-02]
 [ 3.31640800e-03 -2.89073235e-02]]
** Varianza por cada factor:
(array([1.99752121, 1.0182875 ]), array([0.09512006, 0.04848988]), array([0.09512006, 0.14360994]))
```

**Imagen 21.** Resultado de cargas factoriales. *(Elaboración propia, 2024)*

Las cargas factoriales representan la relación entre cada variable y los factores extraídos. Un valor alto nos indica una fuerte relación entre la variable y el factor correspondiente.

Aquí vemos dos factores extraídos. El primer factor explica aproximadamente el 9.51% de la varianza total, mientras que el segundo factor explica el 4.85%. Esto significa que los dos factores juntos explican alrededor del 14.36% de la varianza total en los datos. Esto se obtiene sumando los valores de la varianza explicada por cada factor.

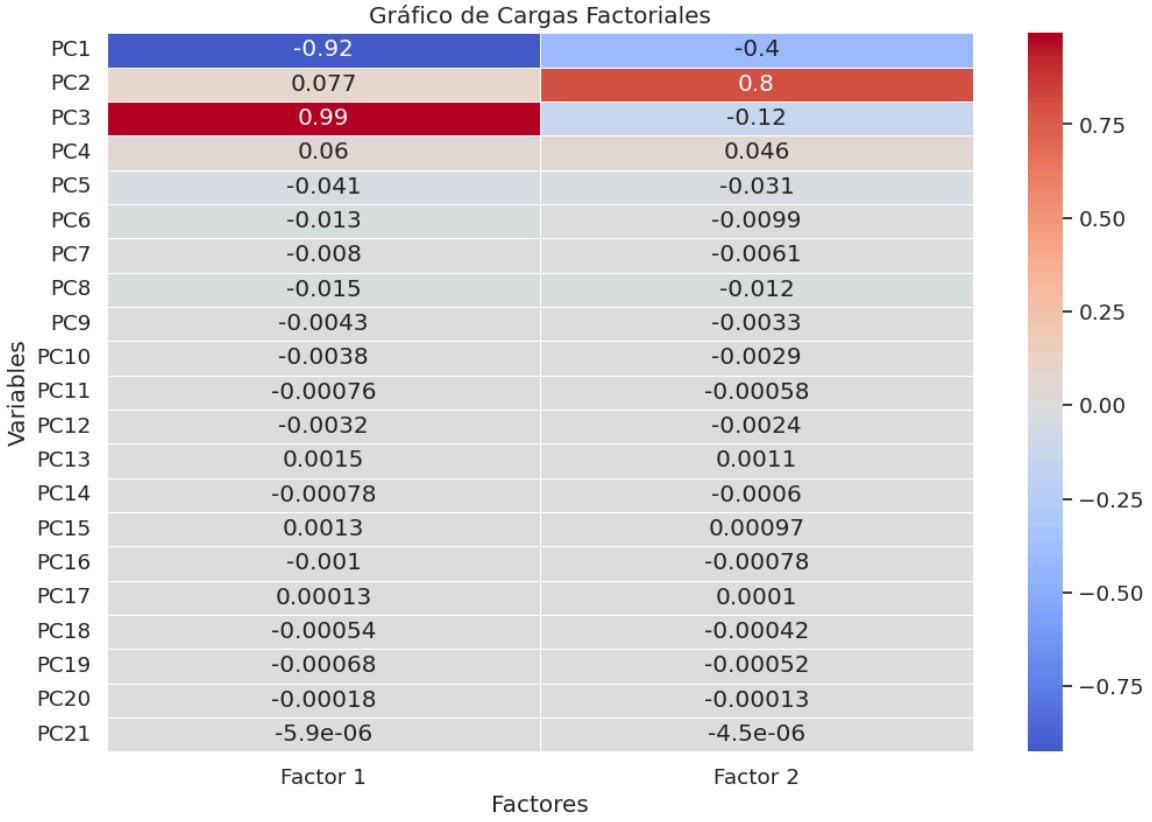


**Imagen 22.** Resultado del gráfico de cargas factoriales. *(Elaboración propia, 2024)*

## 6. Conclusiones

En la entrega anterior del proyecto integrador se inició con el análisis exploratorio de los datos, para poder comprender mejor los registros y cómo se comportan, dado el contexto de la pregunta que deseamos resolver. En nuestro caso, nos enfocamos en las islas de calor con enfoque en la justicia social, para lo cual hemos dispuesto de diferentes orígenes de datos.

Como primera instancia, en términos poblacionales es donde ha sido más exigente el procesamiento de datos, pues era inicialmente necesario unificar todas las entidades, además de limpiar y tratar valores fuera de lo establecido, como lo fueron los asteriscos y valores faltantes. Con este conjunto de datos, la pregunta de negocio que deseábamos responder es si dicho sector poblacional es o no vulnerable; además, el grado de vulnerabilidad dadas sus propias condiciones, para definir un nivel de criticidad. Por tanto, para este punto habíamos ya logrado satisfacer las primeras dos etapas que define la metodología CRISP-ML.

Consecuentemente, continuamos preparando de manera más detallada nuestro conjunto de datos definiendo aquellas variables que nos podrían ayudar a determinar cuando la población se encontrarse en condiciones vulnerables o, inherentemente a su persona podría representar un factor de vulnerabilidad en condiciones desfavorables. Así pues, fue necesario implementar ingeniería de características, hacer escalamiento o normalización y transformaciones, como también explorar la codificación, y emplear diversos métodos de filtrado, tales como obtención de la correlación, análisis de componentes principales o PCA y análisis factorial o FA.

Por tanto, en esta segunda entrega hemos sido capaces de cubrir las primeras tres fases que define la metodología, creando así los cimientos que nos ayudarán a modelar conforme lo establecido, que es segmentar la vulnerabilidad contra un grado de criticidad, lo que posteriormente contrastaremos contra las islas de calor que también se han ido procesando a nivel de AGEB.

## 7. Referencias


Bech, J. (2019). Análisis Multivariado. Universidad Autónoma de Aguascalientes. ISBN 978-607-8652-68-6. https://editorial.uaa.mx/docs/analisis_multivariado.pdf

INEGI. (2020). Sistema de consulta de integración territorial (SCITEL). Principales resultados por AGEB y manzana urbana. INEGI. https://www.inegi.org.mx/app/scitel/Default?ev=10

INEGI. (s. f.). *Publicaciones y mapas*.

https://www.inegi.org.mx/app/biblioteca/ficha.html?upc=889463807469

Kumar Mukhiya, S., y Ahmed, U. (2020). Hands-On Exploratory Data Analysis with Python. Packt Publishing. https://learning.oreilly.com/library/view/hands-on-exploratory-data/9781789537253/0957090f-fa4d-4145-95dd-6d3782e5c04d.xhtml

Mas, J. (2019). Análisis univariante. Universitat Oberta de Catalunya. PID_00268326. https://openaccess.uoc.edu/bitstream/10609/148455/3/AnalisisUnivariante.pdf

Torre, J., *et. al.* (2023). Metodología para identificar y cuantificar islas de calor en entornos urbanos con imágenes satelitales. Centro para el Futuro de las Ciudades, Tecnológico de Monterrey. https://drive.google.com/drive/folders/1p-hPh6o_heBx-HAEKY1CsAioUi1XuRcS?hl=es

Studer, S., *et. al*. (2021). Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology. Preprints 2021, 1, 0. https://doi.org/10.48550/arXiv.2003.05155

Visengeriyeva, L., Kammer, A., Bär, I., Kniesz, A., y Plöd, M. (2023). CRISP-ML(Q). The ML Lifecycle Process. MLOps. INNOQ. https://ml-ops.org/content/crisp-ml

## 8. Anexos

Anexo - Repositorio en GitHub