WBL DERECHOS DE LAS MUJERES

AUTOR: MARIA EUGENIA GONZALEZ

CONTEXTO

WBL son informes semestrales que miden las diferencias de género en la ley. Los puntajes del índice WBL se basan en el promedio de los puntajes de cada economía para los 8 temas incluidos en el puntaje agregado de este año. Una puntuación más alta indica más leyes de igualdad de género, desde la perspectiva de la movilidad, maternidad, pago, emprendedurismo, matrimonio, etc Nos planteamos distintas hipótesis a desarrollar. Desde el Banco Mundial han solicitado un análisis del desarrollo de los Derechos de Mujeres a nivel mundial, para así lograr marcar el campo de acciones presentes y futuras para llegar a una verdadera igualdad.

Preguntas

1. ¿Que países son los que han evolucionado más en el campo de Derechos de Mujeres?
2. ¿Tiene relación la clase social con el Desarrollo de Derechos?
3. Análisis Univariados de Desarrollo de Derechos, relación con la PAGA, MATRIMONIO Y BIENES
4. Con respecto a la movilidad ¿Las mujeres han ganado más derechos?
5. ¿Tienen los hombres y las mujeres los mismos derechos de propiedad sobre los bienes inmuebles?
6. Con respecto a la Libertad Económica ¿Ha habido una evolución favorecedora?
7. ¿La edad a la que hombres y mujeres pueden jubilarse con pensión completa es la misma?
8. ¿Cual es la relación entre la Paga y el Indice WBL?

Objetivo:

En base a todas las variables presentadas en el dataset descubrir: ¿Como han evolucionado los Derechos de las Mujeres?

TRABAJO FINAL DE MARIA EUGENIA GONZALEZ WBL

INTRODUCCION

Contexto empresarial. WBL son informes semestrales que miden las diferencias de género en la ley. Los puntajes del índice WBL se basan en el promedio de los puntajes de cada economía para los 8 temas incluidos en el puntaje agregado de este año. Una puntuación más alta indica más leyes de igualdad de género, desde la perspectiva de la movilidad, maternidad, pago, emprendedurismo, matrimonio, etc Nos planteamos distintas hipotesis a desarrollar. Desde el Banco Mundial han solicitado un análisis del desarrollo de los Derechos de Mujeres a nivel mundial, para así lograr marcar el campo de acciones presentes y futuras para llegar a una verdadera igualdad.

PROBLEMA Y OBJETIVOS

Problema comercial. La tarea consiste en manipular y analizar los datos proporcionados y por medio de visualizaciones responder las preguntas específicas, que se mencionan a continuación.

Preguntas

1. ¿Que paises son los que han evolucionado más en el campo de Derechos de Mujeres?
2. ¿Tiene relación la clase social con el Desarrollo de Derechos?
3. Analisis Univariados de Desarrollo de Derechos, relacion con la PAGA, MATRIMONIO Y BIENES
4. Con respecto a la movilidad ¿Las mujeres han ganado más derechos?
5. ¿Tienen los hombres y las mujeres los mismos derechos de propiedad sobre los bienes inmuebles?
6. Con respecto a la Libertad Economica ¿Ha habido una evolución favorecedora?
7. ¿La edad a la que hombres y mujeres pueden jubilarse con pensión completa es la misma?
8. ¿Cual es la relación entre la Paga y el Indice WBL?

Pregunta Objetivo:

En base a todas las variables presentadas en el dataset descubrir: ¿Como han evolucionado los Derechos de las Mujeres?

DATA ACQUISITION

El dataset elegido contiene datos obtenidos del Banco Mundial, en particular estos datos se centran en la evolucion de ciertos Derechos de las Mujeres, en distintos lugares del mundo. El Dataset es publico y se encuentra en la carpeta de este proyecto.

2.1 Importacion de Librerias

```
import numpy as np
import pandas as pd
import scipy as sp
from prettytable import PrettyTable
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px

mpl.style.use('bmh')

from sklearn.decomposition import PCA
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sm
from sklearn.utils import resample
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import GridSearchCV
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import roc_curve, roc_auc_score
import warnings
warnings.filterwarnings('ignore')
```

2.2 Obtencion de Datos

```
df1= pd.read_excel("/content/drive/MyDrive/Datasetwl.xlsx")   #leer el archivo .
df1.head()
```

	Economy	Economy Code	ISO Code	Region	Income Group	Report Year	WBL INDEX	MOBILITY	Can a woman choose where to live in the same way as a man?
0	Afghanistan	AFG	AFG	South Asia	Low income	1971	26.25	25	No
1	Afghanistan	AFG	AFG	South Asia	Low income	1972	26.25	25	No
2	Afghanistan	AFG	AFG	South Asia	Low income	1973	26.25	25	No
3	Afghanistan	AFG	AFG	South Asia	Low income	1974	26.25	25	No
4	Afghanistan	AFG	AFG	South Asia	Low income	1975	26.25	25	No

5 rows × 55 columns

** EDA (análisis univariado, bivariado y multivariado)**

```
def df1_explore(df1):
    print( 'Shape: ', df1.shape)

    t = PrettyTable(['Column',
                     'Type',
                     'Non-Null',
                     'Nulls',
                     'Unique',
                     'Example',
                    ])
```

```
    for c in df1.columns:
        t.add_row([c,
                   df1[c].dtype,
                   len(df1[c])-np.sum(df1[c].isna()),
                   np.sum(df1[c].isna()),
                   np.count_nonzero(df1[c].unique()),
                   df1[~df1[c].isnull()][c].iloc[0],
                  ])
    print(t)
    print()
    return

df1_explore(df1)
```

```
|                                        Column
+---------------------------------------------------------------------------
|                                       Economy
|                                     Economy Code
|                                       ISO Code
|                                        Region
|                                     Income Group
|                                     Report Year
|                                      WBL INDEX
|                                       MOBILITY
|                   Can a woman choose where to live in the same way as a man?
|                 Can a woman travel outside her home in the same way as a man?
|                   Can a woman apply for a passport in the same way as a man?
|                Can a woman travel outside the country in the same way as a ma
|                                      WORKPLACE
|                        Can a woman get a job in the same way as a man?
|              Does the law prohibit discrimination in employment based on gend
|                    Is there legislation on sexual harassment in employment?
|  Are there criminal penalties or civil remedies for sexual harassment in
|                                         PAY
|                Does the law mandate equal remuneration for work of equal valu
|                      Can a woman work at night in the same way as a man?
|              Can a woman work in a job deemed dangerous in the same way as a
|                Can a woman work in an industrial job in the same way as a man
|                                       MARRIAGE
| Is the law free of legal provisions that require a married woman to obey
|                   Can a woman be head of household in the same way as a man?
|                Is there legislation specifically addressing domestic violence
|               Can a woman obtain a judgment of divorce in the same way as a m
|                     Does a woman have the same rights to remarry as a man?
|                                      PARENTHOOD
|                    Is paid leave of at least 14 weeks available to mothers?
|                                Length of paid maternity leave
|                Does the government administer 100% of maternity leave benefit
|                          Is there paid leave available to fathers?
|                                Length of paid paternity leave
|                                Is there paid parental leave?
|                                     Shared days
|                                  Days for the mother
```

```
|                                  Days for the mother
|                                  Days for the father
|                        Is dismissal of pregnant workers prohibited?
|                                    ENTREPRENEURSHIP
|       Does the law prohibit discrimination in access to credit based on g
|                Can a woman sign a contract in the same way as a man?
|               Can a woman register a business in the same way as a man?
|               Can a woman open a bank account in the same way as a man?
|                                         ASSETS
|          Do men and women have equal ownership rights to immovable proper
|     Do sons and daughters have equal rights to inherit assets from their
|       Do male and female surviving spouses have equal rights to inherit a
| Does the law grant spouses equal administrative authority over assets dur
|          Does the law provide for the valuation of nonmonetary contributi
|                                        PENSION
|   Is the age at which men and women can retire with full pension benefits
|  Is the age at which men and women can retire with partial pension benefi
|              Is the mandatory retirement age for men and women the same?
|       Are periods of absence due to childcare accounted for in pension be
+--------------------------------------------------------------------------
```

```
df1.duplicated().any()
```

```
False
```

ANALISIS No se variables de datos nulas ni duplicadas

Verifico los valores únicos de las variables de tipo object

```
def df1_unique_val_col(df1, col_list = [] ):

    cant_table_col = 0
    for i in col_list:
        l = len(df1[i].unique())
        if cant_table_col < l:
            cant_table_col = l

    A = []
    for i in range(cant_table_col):
        A.append('Value : '+str(i))

    dg = pd.DataFrame(index = A,
                      columns = df[col_list].columns)

    for m in df1[col_list]:
        uni = df1[m].unique()
        le = len(uni)

        for j in range(cant_table_col):
            if j < le:
                dg[m][j] = uni[j]
            else:
                dg[m][j] = '-'

    print('Valores únicos de algunos campos de interes')
    return dg
```

Estadisticas

```
df1.describe().round(2).T
```

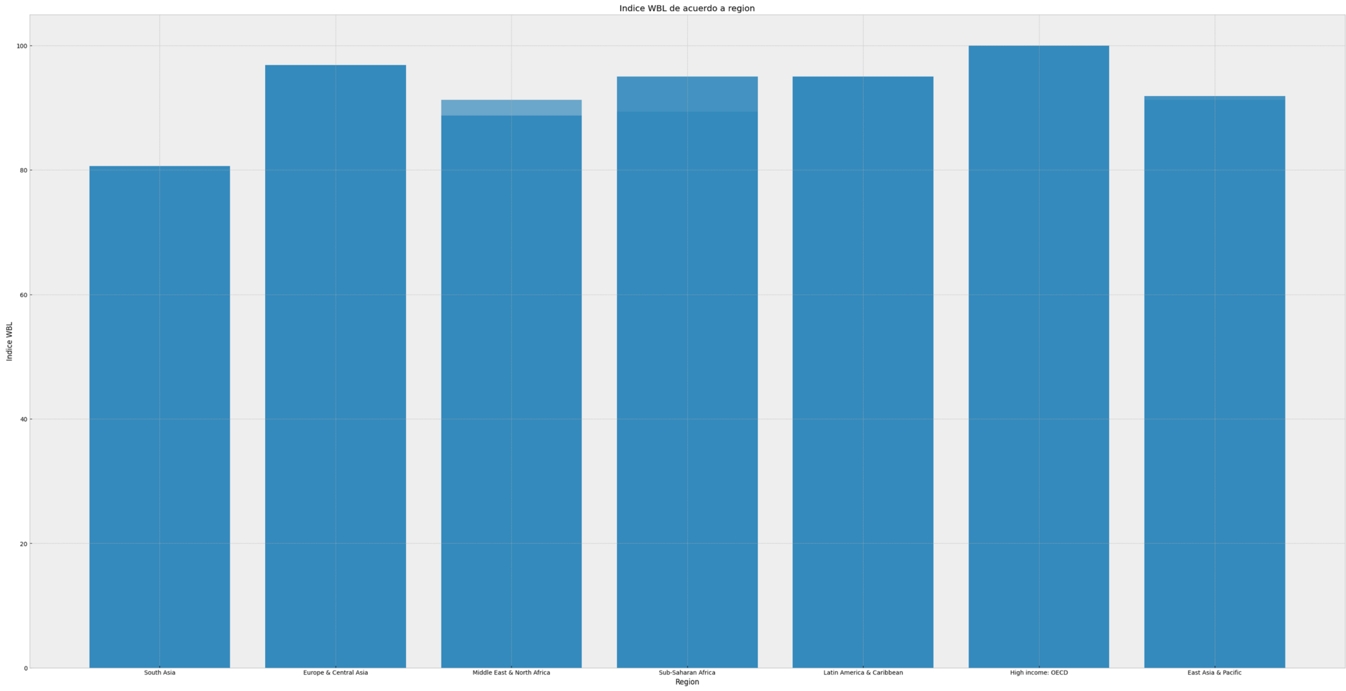
	count	mean	std	min	25%	50%	75%	m
Report Year	10070.0	1997.00	15.30	1971.0	1984.00	1997.00	2010.00	202
WBL INDEX	10070.0	59.73	18.62	17.5	46.88	59.38	73.12	10
MOBILITY	10070.0	82.14	25.73	0.0	75.00	100.00	100.00	10
WORKPLACE	10070.0	43.08	33.70	0.0	25.00	25.00	75.00	10
PAY	10070.0	47.66	31.27	0.0	25.00	50.00	75.00	10
MARRIAGE	10070.0	61.81	29.89	0.0	40.00	80.00	80.00	10
PARENTHOOD	10070.0	34.98	30.64	0.0	0.00	20.00	60.00	10
Length of paid maternity leave	10070.0	85.44	62.11	0.0	60.00	84.00	101.00	63
Length of paid paternity leave	10070.0	1.75	6.84	0.0	0.00	0.00	0.00	18
Shared days	10070.0	38.56	159.93	0.0	0.00	0.00	0.00	146
Days for the mother	10070.0	5.27	40.54	0.0	0.00	0.00	0.00	109
Days for the father	10070.0	3.68	27.85	0.0	0.00	0.00	0.00	36
ENTREPRENEURSHIP	10070.0	72.72	21.23	0.0	75.00	75.00	75.00	10
ASSETS	10070.0	73.77	29.20	0.0	40.00	80.00	100.00	10

Analizo distribuciones y participaciones

1.¿Que países son los que han evolucionado más en el campo de Derechos de Mujeres?

```
fig, ax = plt.subplots(figsize=(40,20))
ax.bar ( df1['Region'],df1['WBL INDEX'], alpha=0.7)
ax.set_title('Indice WBL de acuerdo a region')
ax.set_ylabel('Indice WBL')
ax.set_xlabel('Region')
```

```
Text(0.5, 0, 'Region')
```

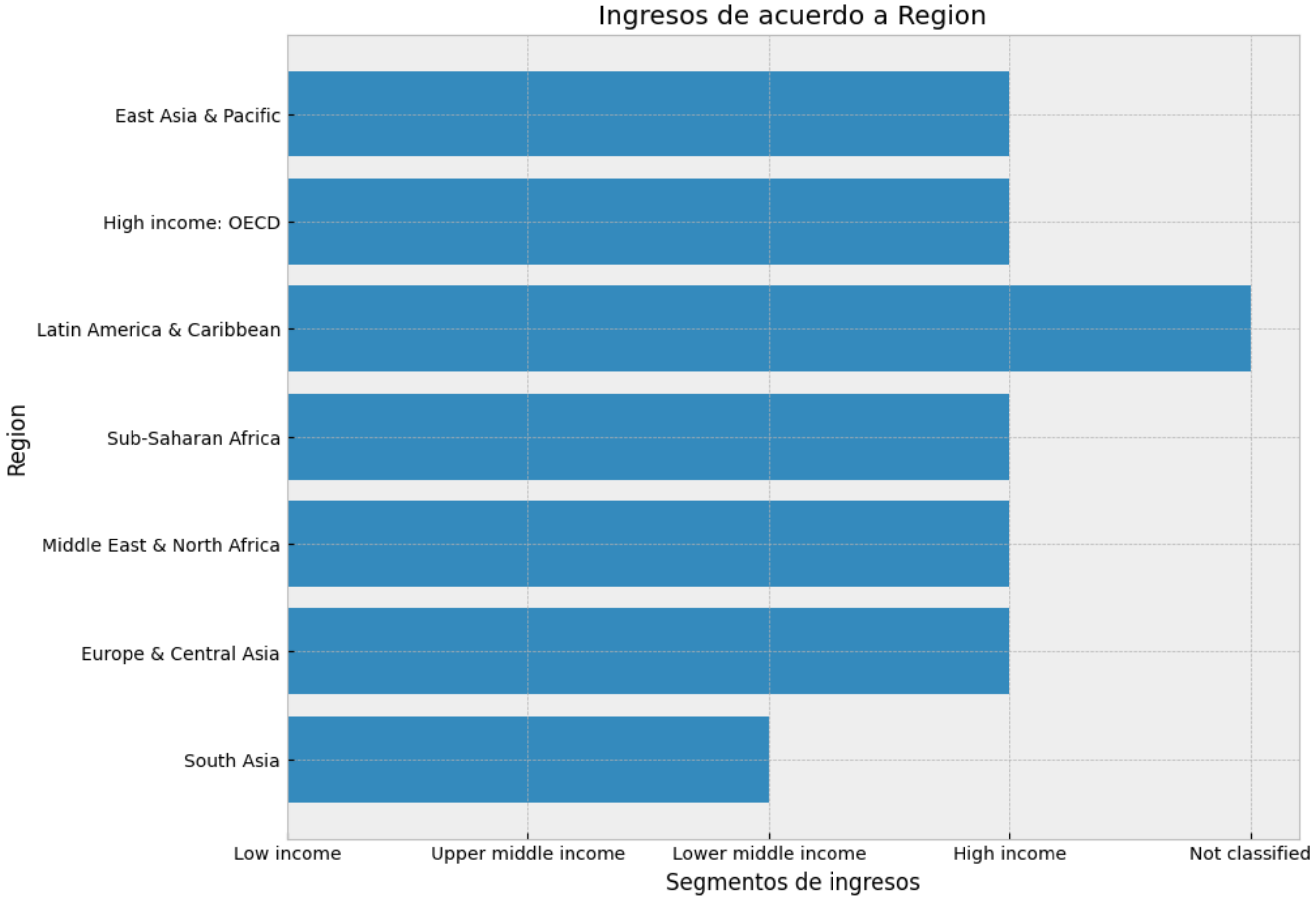


ANALISIS En esta grafica se denota la diferencia entre regiones mas desarolladas en el campo de los derechos de mujeres. Podemos ver que Europa y Asia central, además de los paises integrantes de grandes ingresos OECD cuentan con mayor desarrollo que los demás paises.

2.¿Tiene relación la clase social con el Desarrollo de Derechos?

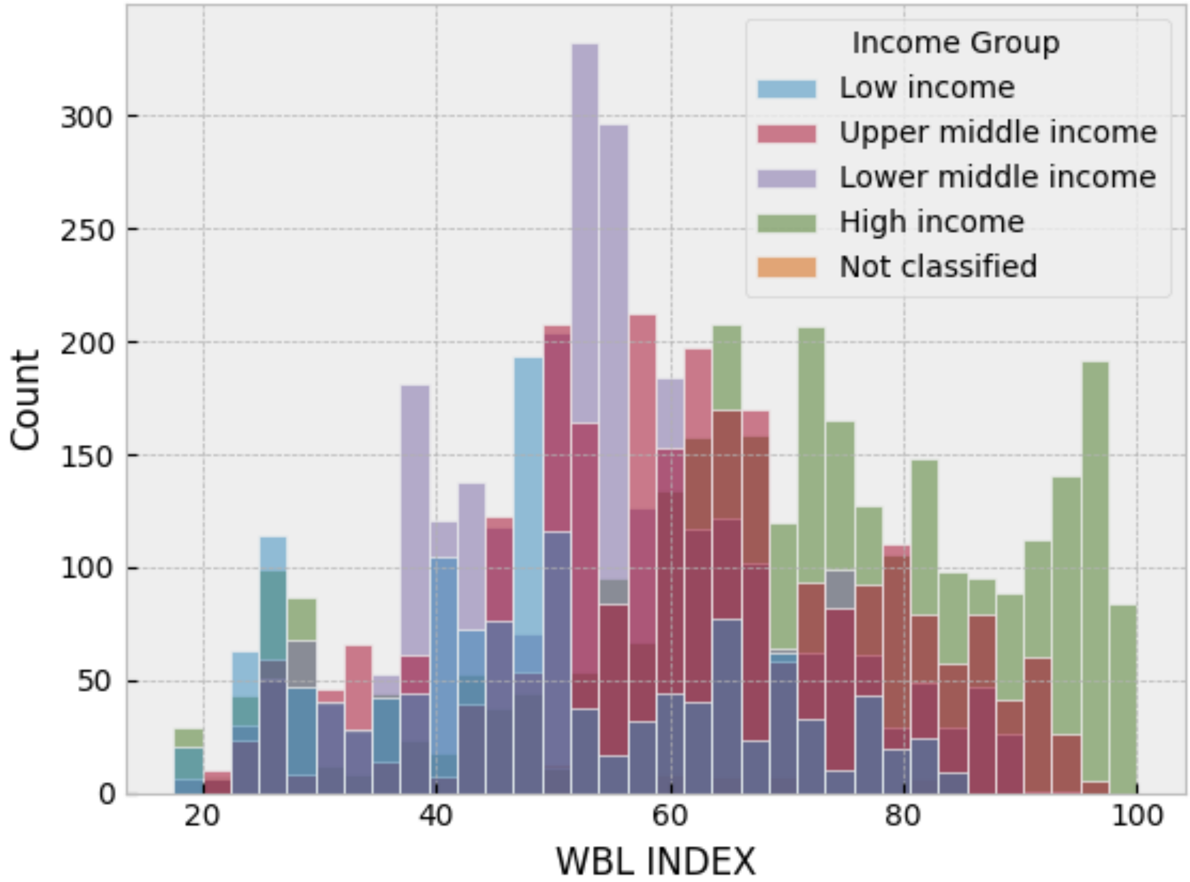
```
fig, ax = plt.subplots(figsize=(10, 8))

ax.barh(df1['Region'], df1['Income Group'])
ax.set_title('Ingresos de acuerdo a Region')
ax.set_ylabel('Region')
ax.set_xlabel('Segmentos de ingresos')
```

```
Text(0.5, 0, 'Segmentos de ingresos')
```



```
sns.histplot(x = df1['WBL INDEX'], hue = df1['Income Group'], data= df1)
```

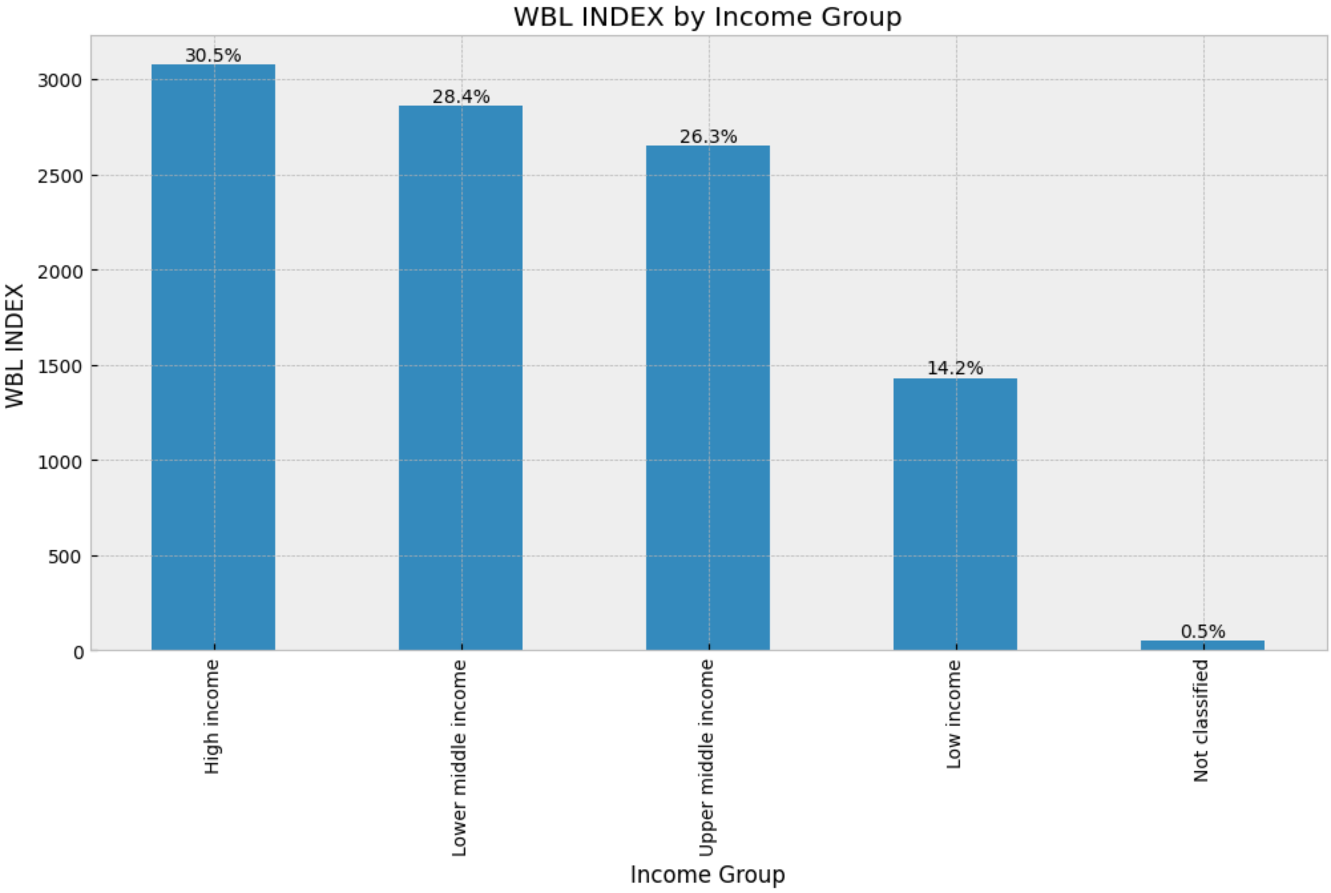
```
<Axes: xlabel='WBL INDEX', ylabel='Count'>
```



```
def column_exploration(df1,column_name,sort_index=False):

    print(f'COLUMN: {column_name}')
    abs_values =df1[column_name].value_counts()
    prc_values =(df1[column_name].value_counts(normalize=True)*100).apply(lambda
    df1_values = pd.merge(abs_values, prc_values, left_index=True, right_index=T

    if sort_index is True:
        df1_values = df1_values.sort_index(axis = 0)

    ax = df1_values[column_name+'_abs'].plot(kind='bar', figsize=(12,6))
    ax.set_title(f"WBL INDEX by {column_name}")
    ax.set_xlabel(column_name)
    ax.set_ylabel('WBL INDEX')
    rects = ax.patches
    labels = [f'{p}%' for p in df1_values[column_name+'_prc'].to_list()]
    for rect, label in zip(rects, labels):
        height = rect.get_height()
        ax.text(rect.get_x() + rect.get_width()/2 , height + 1, label,
                ha='center', va='bottom')

    return
```

```
column_exploration(df1,'Income Group')
```

```
COLUMN: Income Group
```

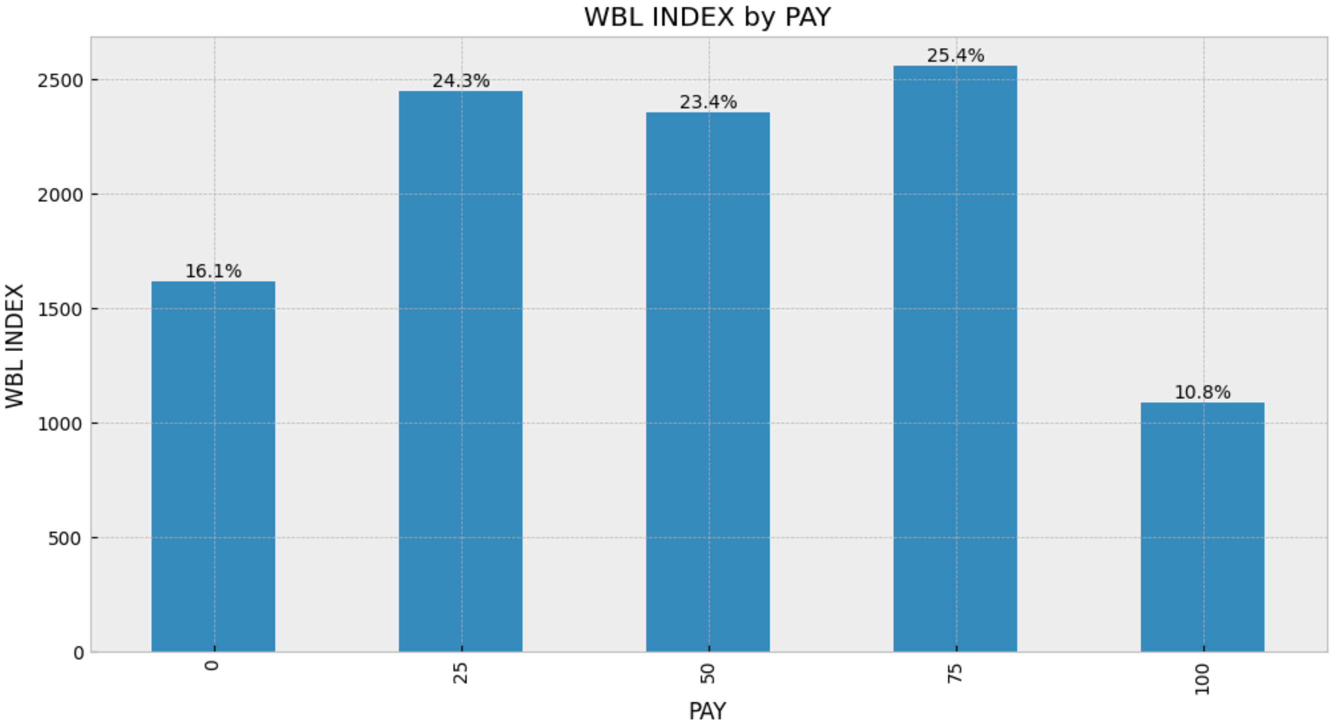


ANALISIS Si bien en la primera grafica podemos notar que no existe una manera de afirmar si hay una verdadera relación entre clase social y desarrollo, sin cierto error, ya que Latinoamerica y el caribe no estan clasificados. Por medio de una segunda y tercera lectura notamos que si hay relación directa con el nivel de ingresos y el desarrollo de Derechos femeninos, así los grupos que más desarrollados se encuentran son grupos de clases de ingresos altos y medios inclinados a altos.

3.Analisis Univariados de Desarrollo de Derechos, relacion con la PAGA, MATRIMONIO Y BIENES

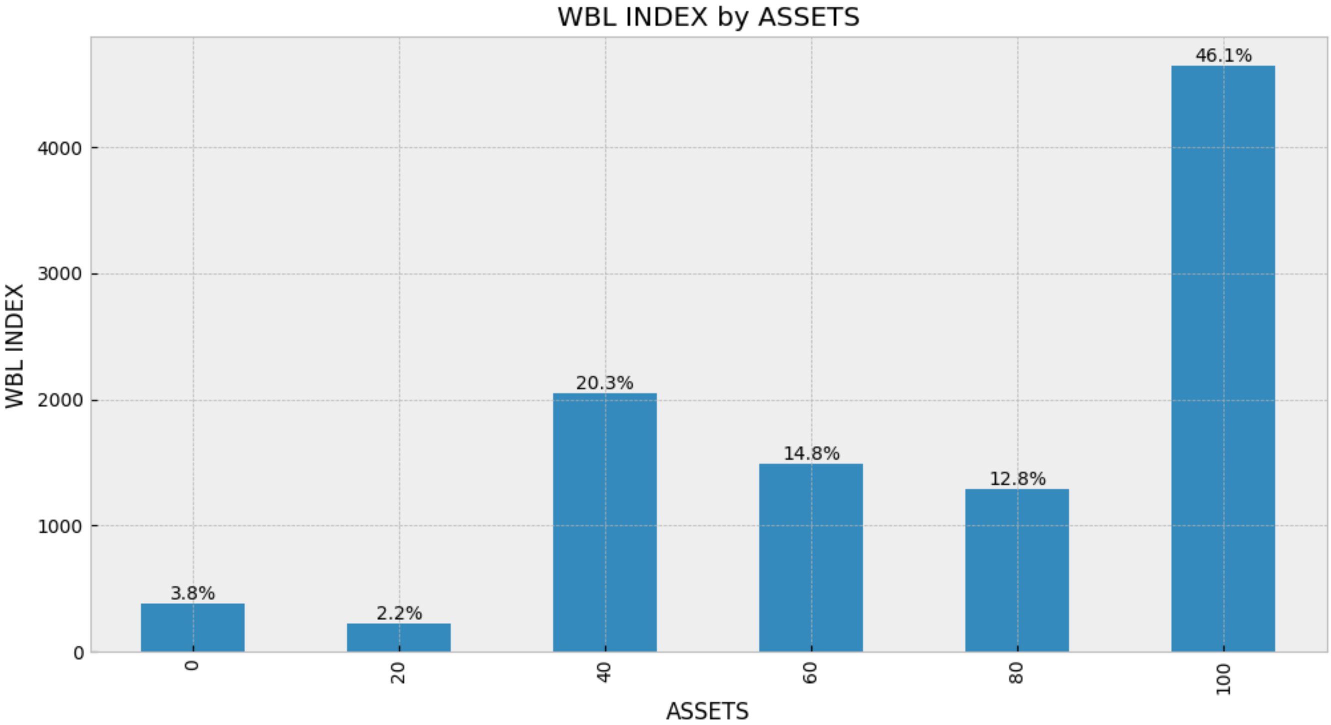
```
column_exploration(df1,'PAY',True)
```

```
COLUMN: PAY
```



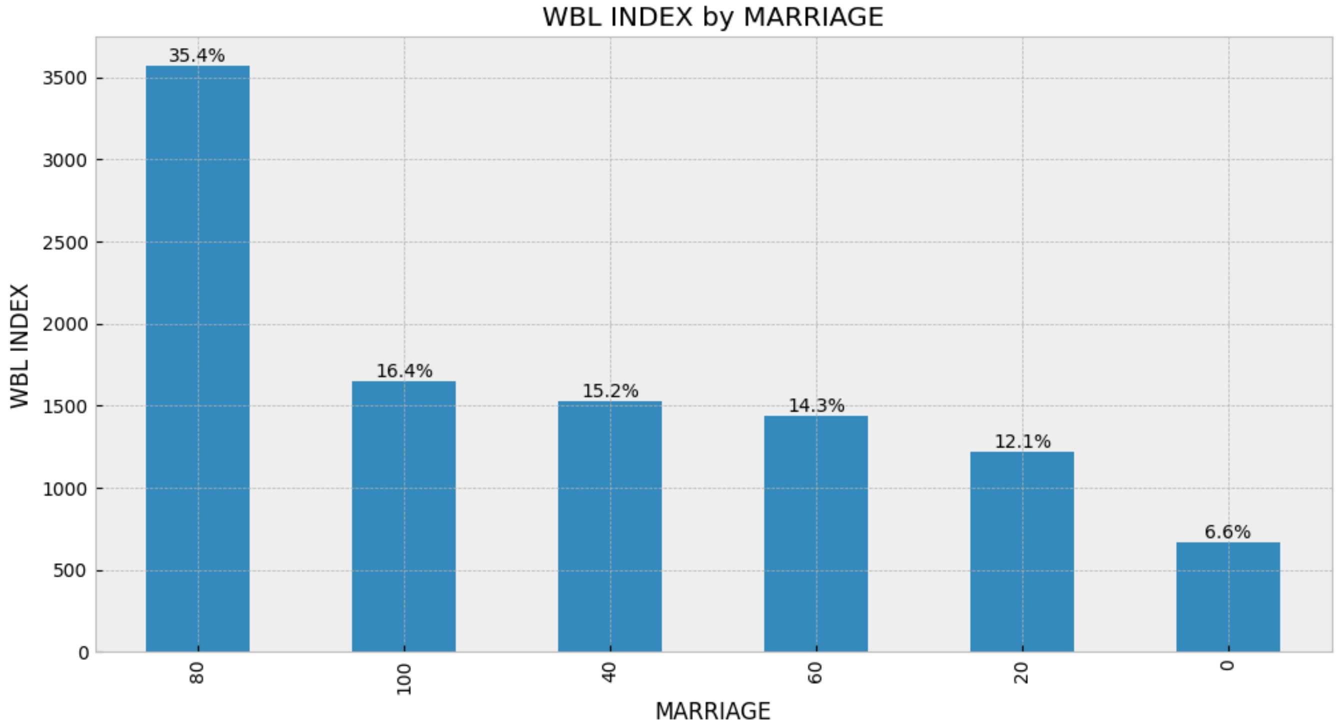
```
column_exploration(df1,'ASSETS',True)
```

```
COLUMN: ASSETS
```



```
column_exploration(df1,'MARRIAGE')
```

```
COLUMN: MARRIAGE
```



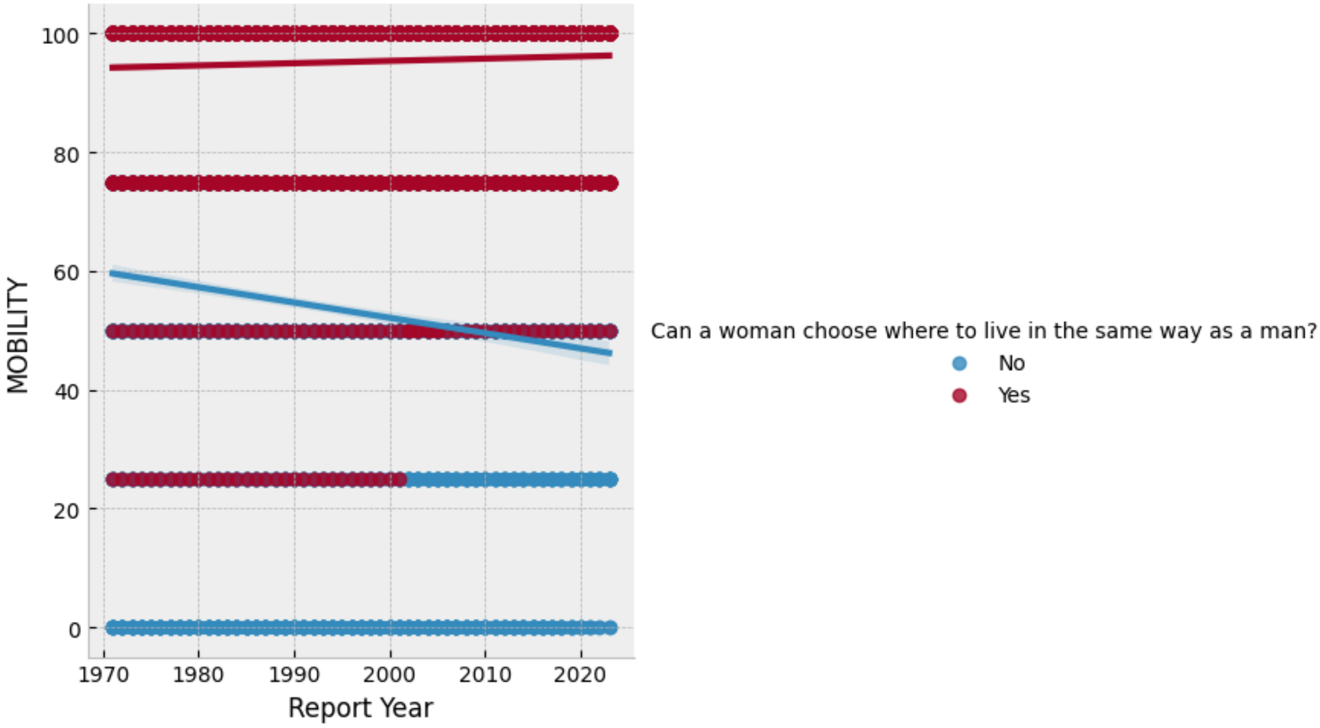
ANALISIS Podemos notar que no hay una relacion directa entre paga y la evolucion de Derechos, pero si en lo relativo al matrimonio y bienes.

```
from seaborn import lmplot
```

4.Con respecto a la movilidad ¿Las mujeres han ganado más derechos?

```
lmplot(x="Report Year", y="MOBILITY", hue= "Can a woman choose where to live in
```

```
<seaborn.axisgrid.FacetGrid at 0x79233ba80940>
```

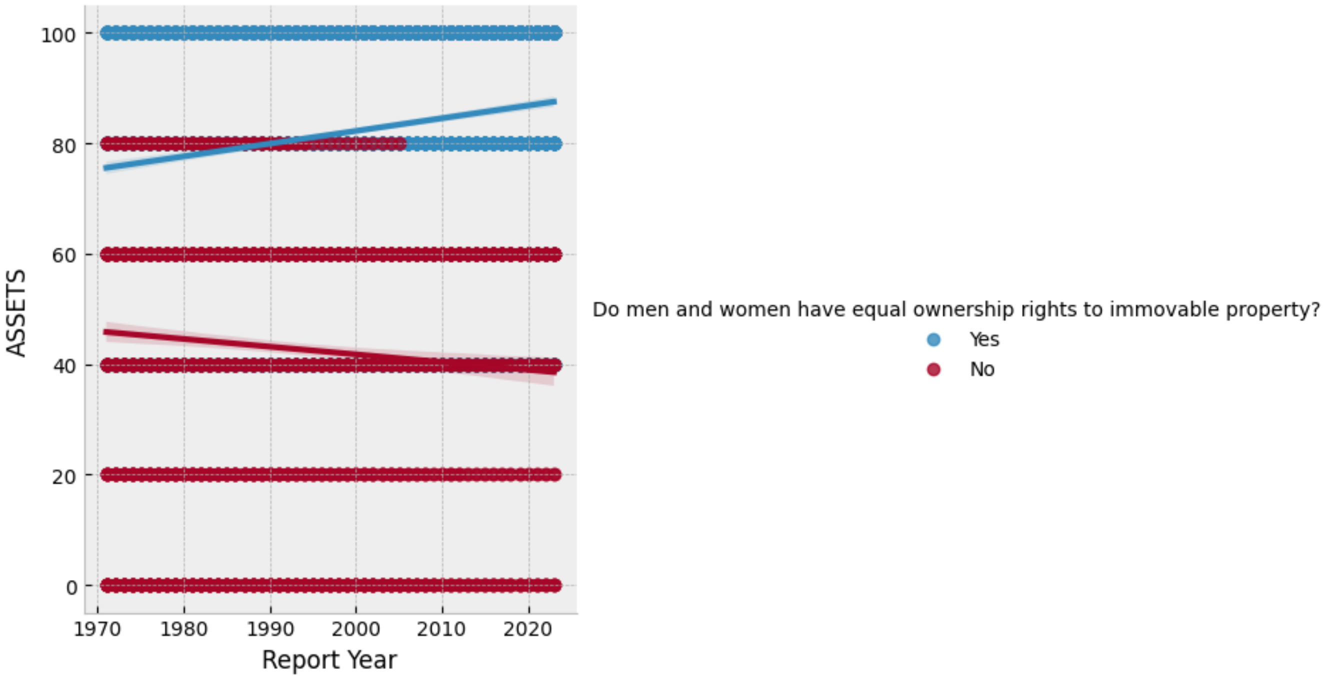


ANALISIS Tal como se puede ver, la respuesta más que con una variable de tiempo tiene que ver con la movilidad con la que cuentan estas mujeres. Así en grupos con menor movilidad la respuesta es negativa, inversamente de lo que sucede en grupos de más movilidad. Sin embargo se nota que hay aumento de libertad a partir de 1990, por lo que afirmamos que dependiendo la movilidad la respuesta será afirmativa o negativa.

5.¿Tienen los hombres y las mujeres los mismos derechos de propiedad sobre los bienes inmuebles?

```
lmplot(x="Report Year", y="ASSETS", hue="Do men and women have equal ownership r
```

```
<seaborn.axisgrid.FacetGrid at 0x79233c8faad0>
```

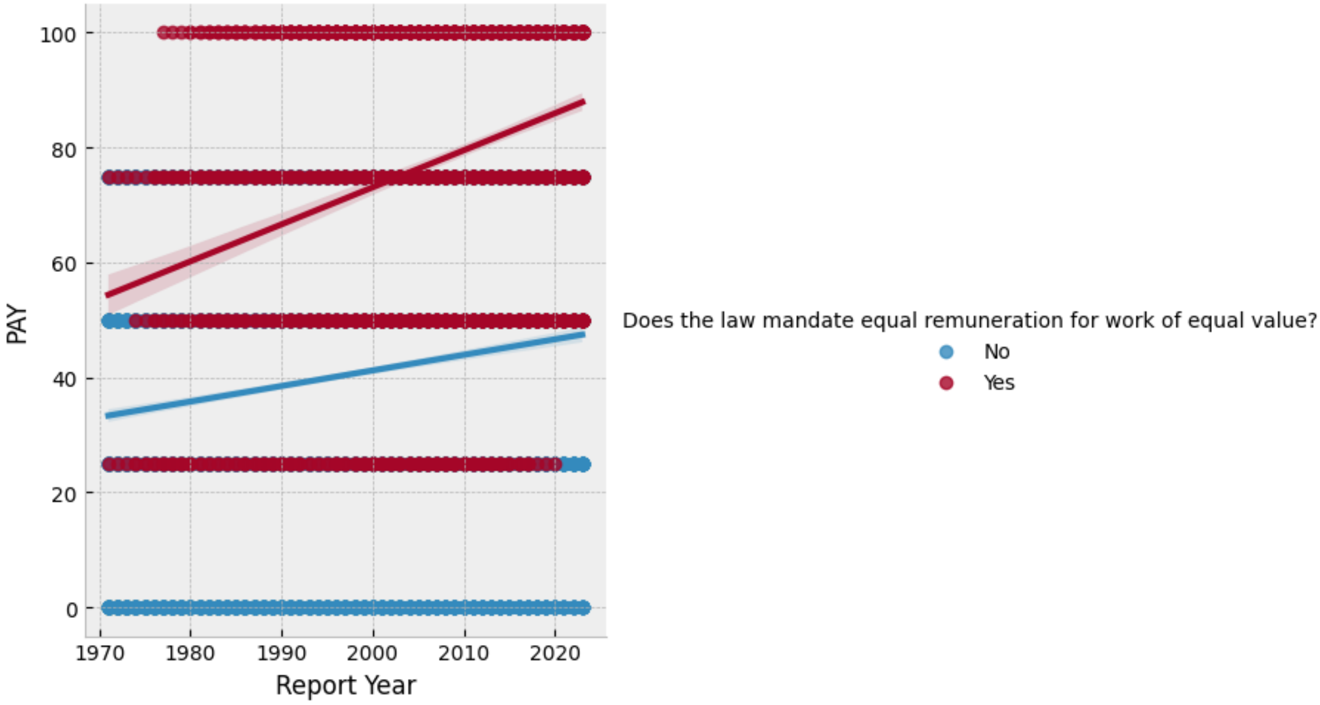


ANALISIS Al igual que en la pregunta anterior, la respuesta más que con una variable de tiempo tiene que ver con la movilidad con la que cuentan estas mujeres. A partir del 2000 solo en segmentos desarrollados altos las mujeres afirman tener igualdad, sin embargo en la mayoria de los segmentos e historicamente sigue siendo un sector para tomar acciones y alcanzar la igualdad.

6.Con respecto a la Libertad Economica ¿Ha habido una evolución favorecedora?

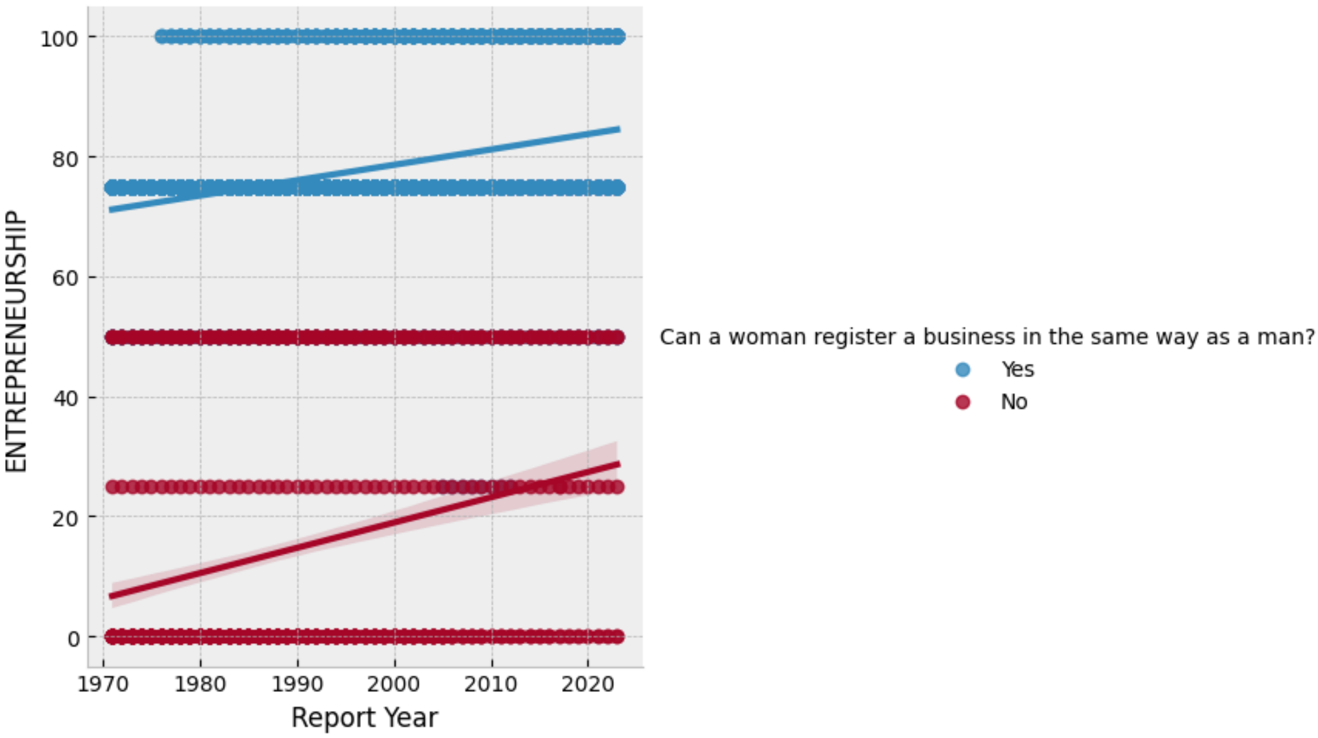
```
lmplot(x="Report Year", y="PAY", hue="Does the law mandate equal remuneration fo
```

```
<seaborn.axisgrid.FacetGrid at 0x79233cba0ee0>
```



```
lmplot(x="Report Year", y="ENTREPRENEURSHIP", hue="Can a woman register a busine
```

```
<seaborn.axisgrid.FacetGrid at 0x79233ca99c90>
```

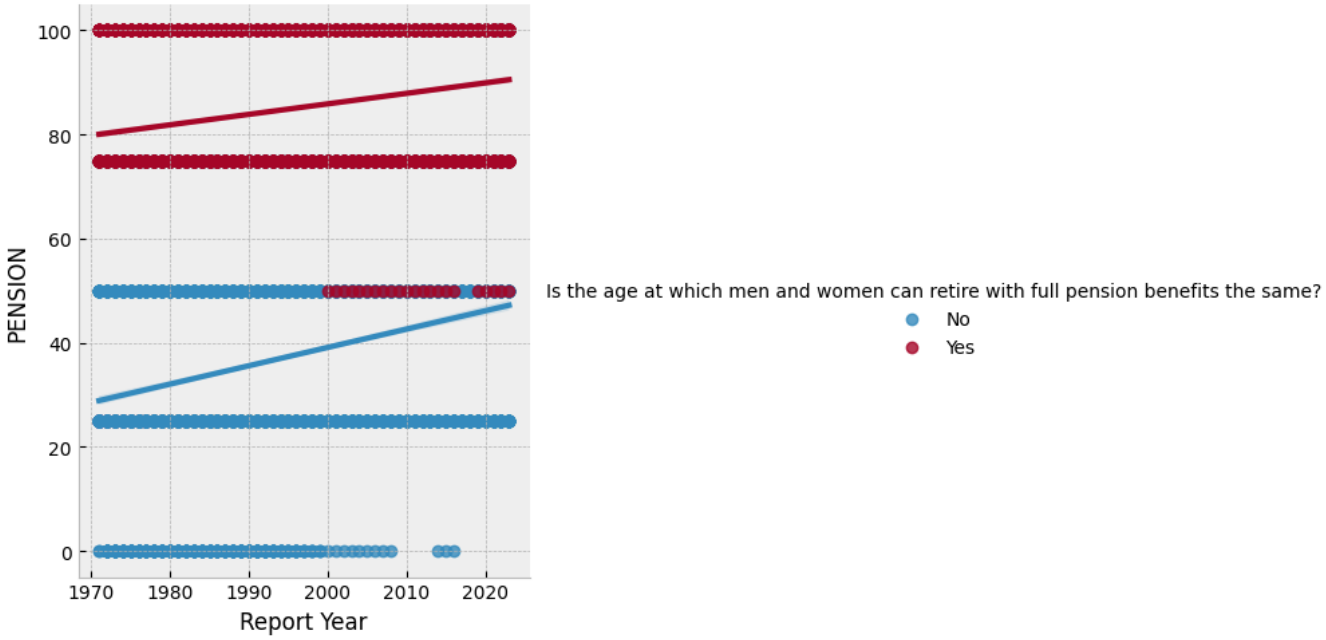


ANALISIS Bajo el análisis de dos puntos, si hubo una evolución favorecedora de libertades pero que ha beneficiados más a mujeres dependientes de alguna empresa, mientras que en mujeres emprendedoras independiente no hubo una evolución historica favorecedora, si excepcionalmente en casos de desarrollo alto.

7.¿La edad a la que hombres y mujeres pueden jubilarse con pensión completa es la misma?

```
lmplot(x="Report Year", y="PENSION", hue="Is the age at which men and women can
```

```
<seaborn.axisgrid.FacetGrid at 0x79233d7615a0>
```



ANALISIS La respuesta es parcialmente afirmativa, ya que también depende de que sector se trate. En el 20% correspondiente a los menos desarrollados la edad continua siendo distinta.

8.¿Cual es la relacion entre la Paga y el Indice WBL?

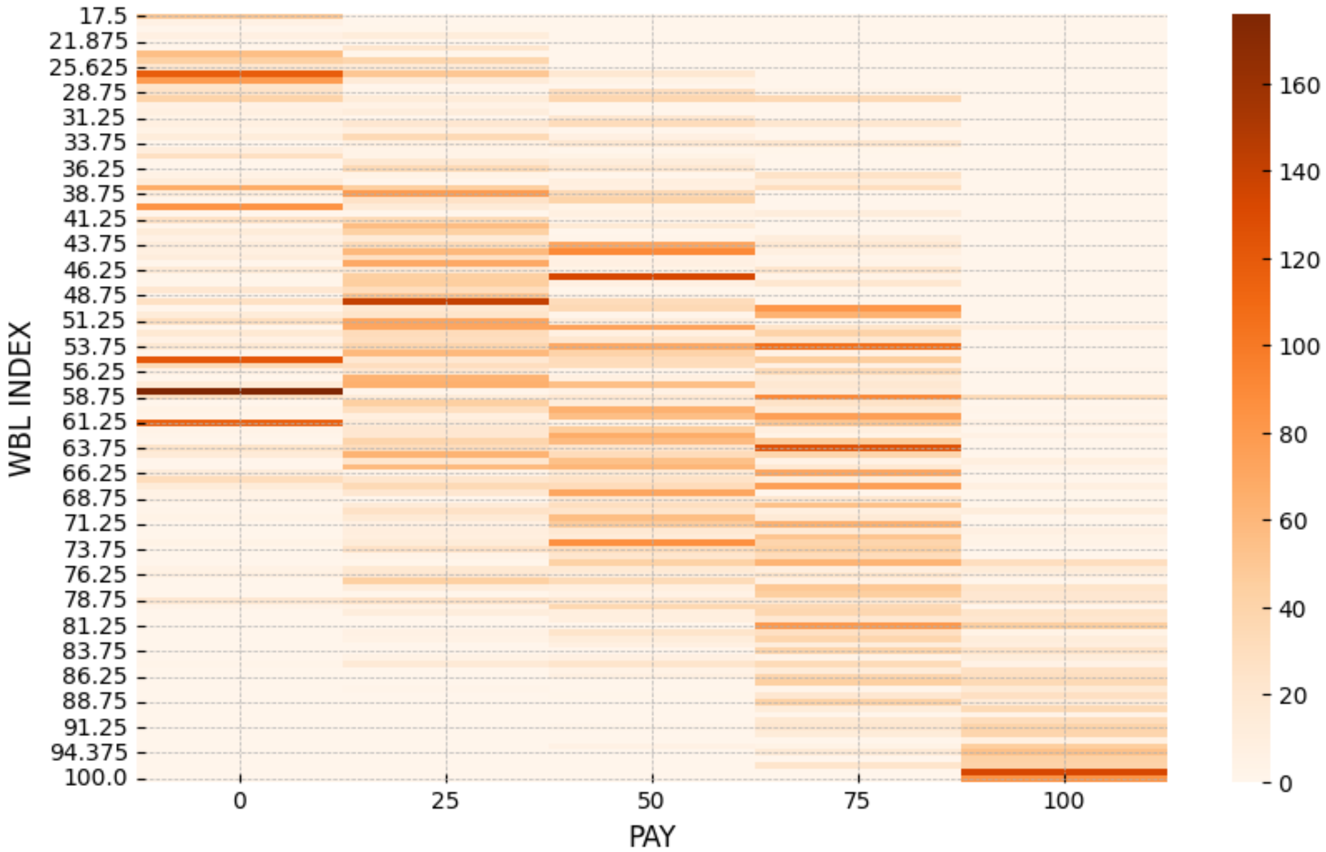
```
crosstab=pd.crosstab(index=df1['WBL INDEX'], columns=df1['PAY'])
crosstab
```

PAY	0	25	50	75	100
WBL INDEX					
17.500	49	0	0	0	0
18.750	6	0	0	0	0
20.000	0	1	0	0	0
20.625	7	13	0	0	0
21.875	2	0	0	0	0
...	...	...	...	...	...
94.375	0	0	0	16	54
95.000	0	0	0	0	42
96.875	0	0	0	23	41
97.500	0	0	0	0	133
100.000	0	0	0	0	84

121 rows × 5 columns

```
from matplotlib import rcParams
plt.subplots(figsize=(10,6))
rcParams['figure.figsize'] = 8,4
import seaborn as sns
sns.heatmap(crosstab,cmap='Oranges')
```

```
<Axes: xlabel='PAY', ylabel='WBL INDEX'>
```

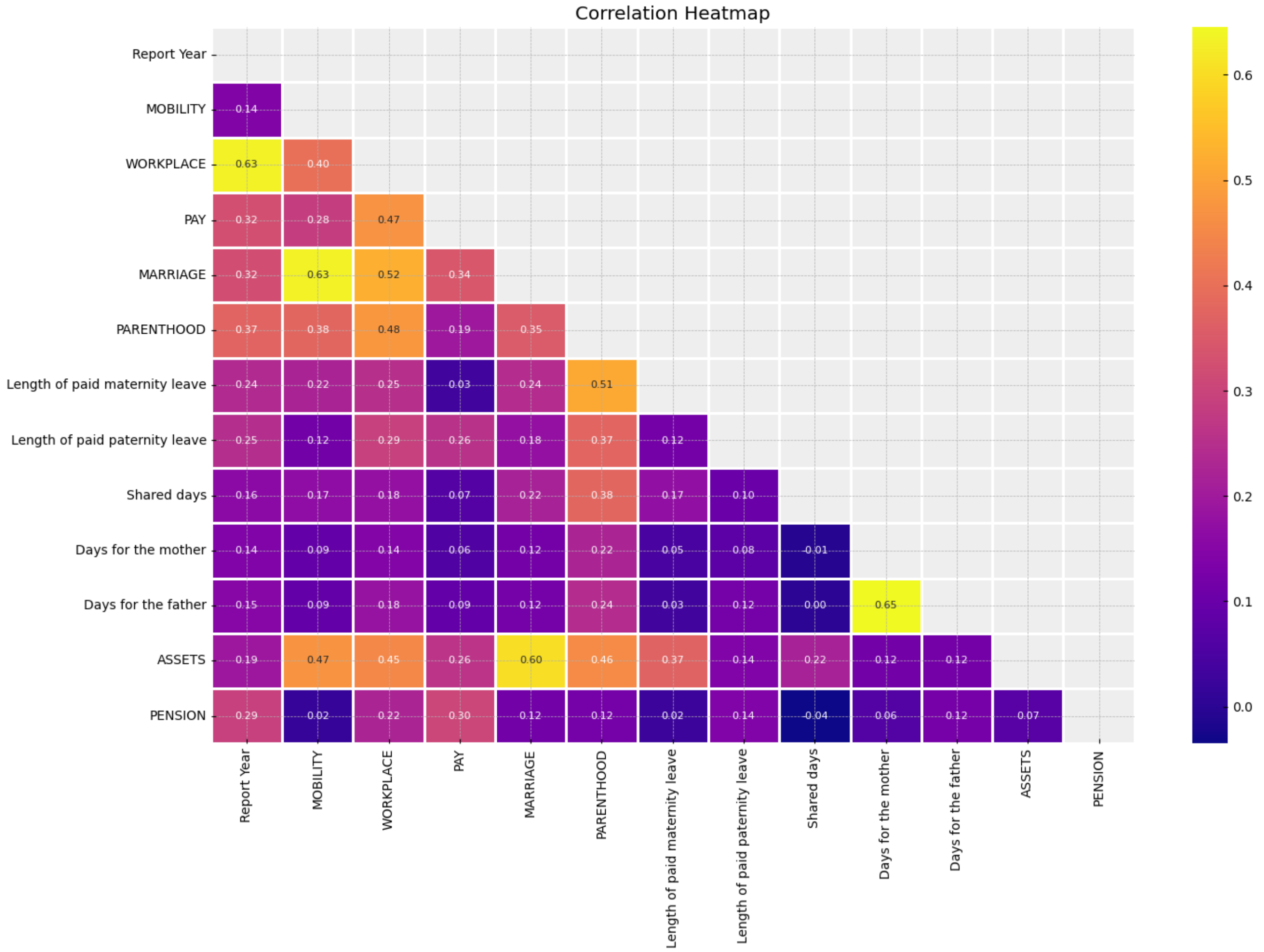


ANALISIS En esta situación lo que podemos ver es que la mayor concentración se encuentra en los sectores intermedios en la franja de 43 y 58, mostrandonos que no existe relacion entre la paga y la evolucion de Derechos

9.Analisis Multivariado

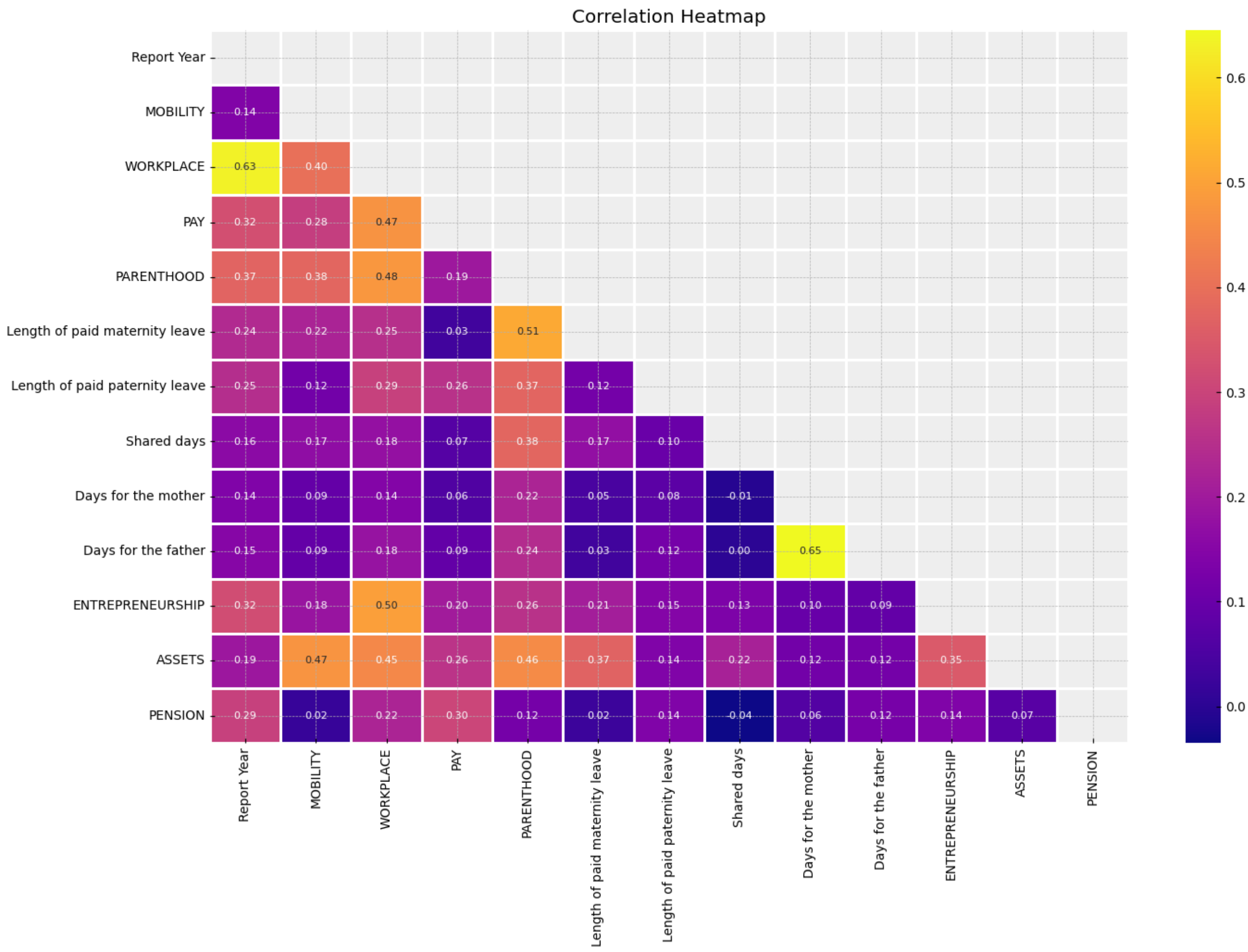
9.1 Correlacion Indice vs Emprendedurismo

```
#plt.figure(dpi = 90,figsize= (10,10))
df1_ = df1.drop(['WBL INDEX','ENTREPRENEURSHIP'],axis=1)

plt.figure(figsize= (16,10))
mask = np.triu(np.ones_like(df1_.corr(),dtype = bool))
sns.heatmap(df1_.corr(),mask = mask, fmt = ".2f",annot=True,lw=1,cmap = 'plasma'
plt.yticks(rotation = 0)
plt.xticks(rotation = 90)
plt.title('Correlation Heatmap')
plt.show()
```



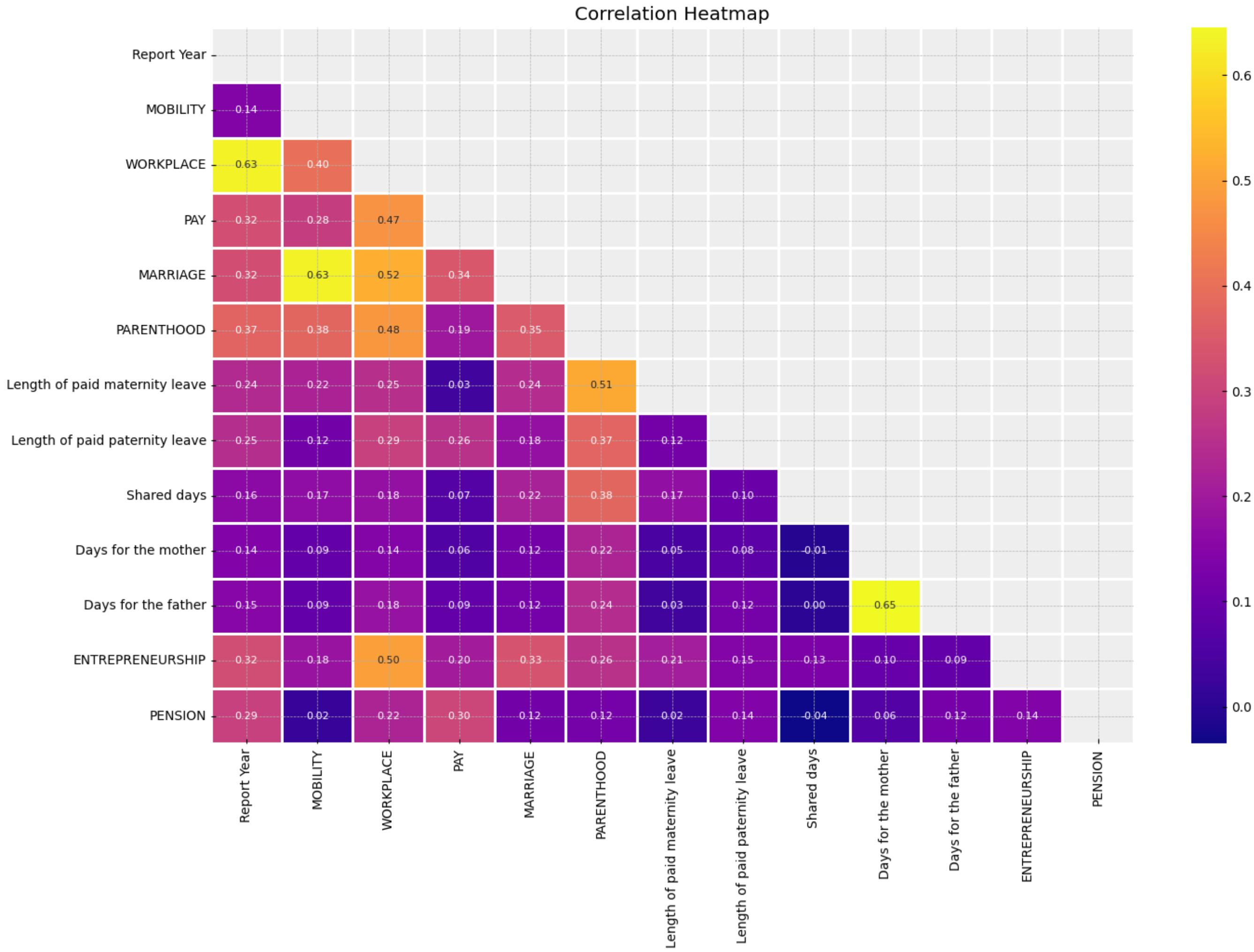
9.2 Correlacion Indice vs Matrimonio

```
#plt.figure(dpi = 90,figsize= (10,10))
df1_ = df1.drop(['WBL INDEX','MARRIAGE'],axis=1)

plt.figure(figsize= (16,10))
mask = np.triu(np.ones_like(df1_.corr(),dtype = bool))
sns.heatmap(df1_.corr(),mask = mask, fmt = ".2f",annot=True,lw=1,cmap = 'plasma'
plt.yticks(rotation = 0)
plt.xticks(rotation = 90)
plt.title('Correlation Heatmap')
plt.show()
```



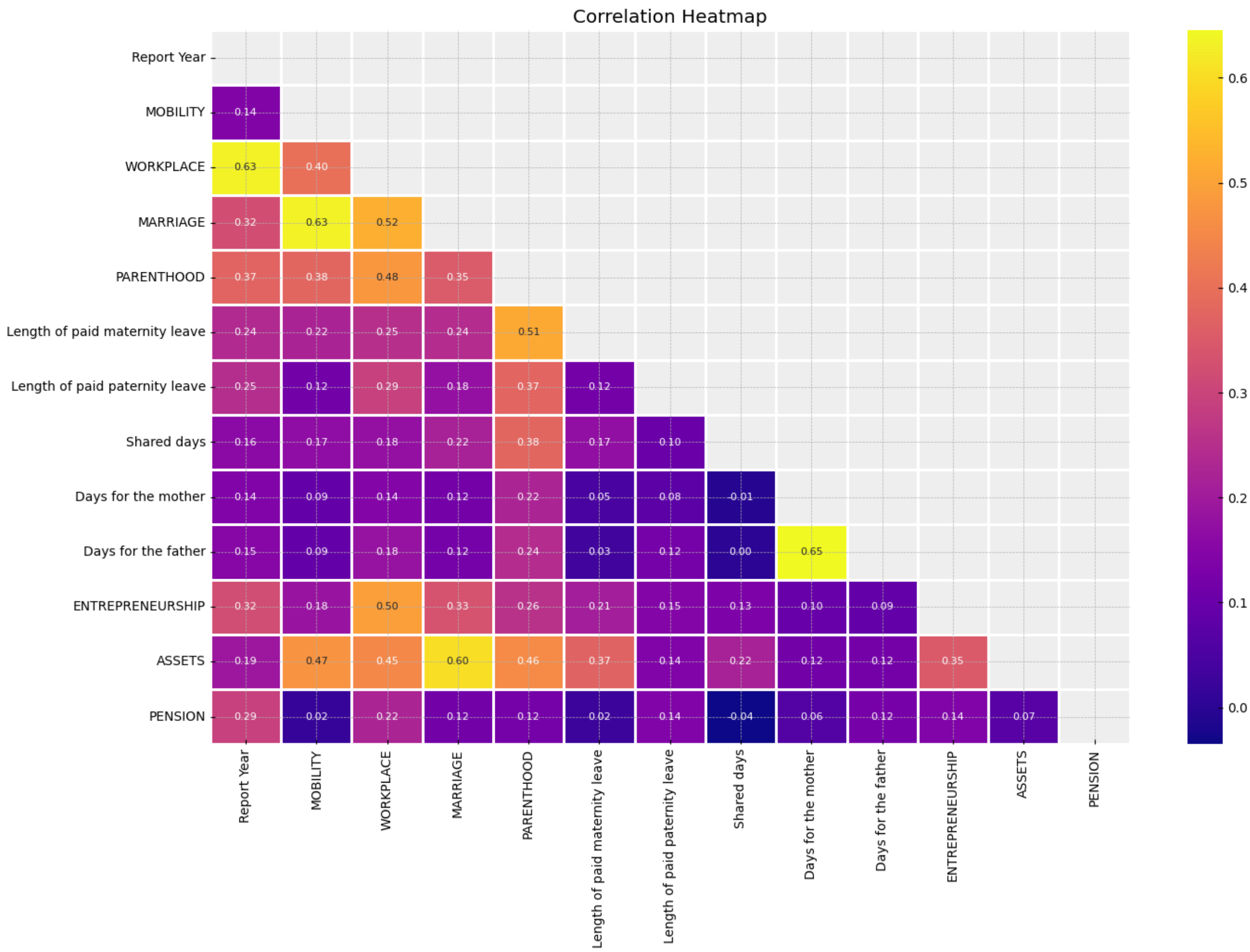
9.3 Correlacion Indice vs Bienes

```
#plt.figure(dpi = 90,figsize= (10,10))
df1_ = df1.drop(['WBL INDEX','ASSETS'],axis=1)

plt.figure(figsize= (16,10))
mask = np.triu(np.ones_like(df1_.corr(),dtype = bool))
sns.heatmap(df1_.corr(),mask = mask, fmt = ".2f",annot=True,lw=1,cmap = 'plasma'
plt.yticks(rotation = 0)
plt.xticks(rotation = 90)
plt.title('Correlation Heatmap')
plt.show()
```



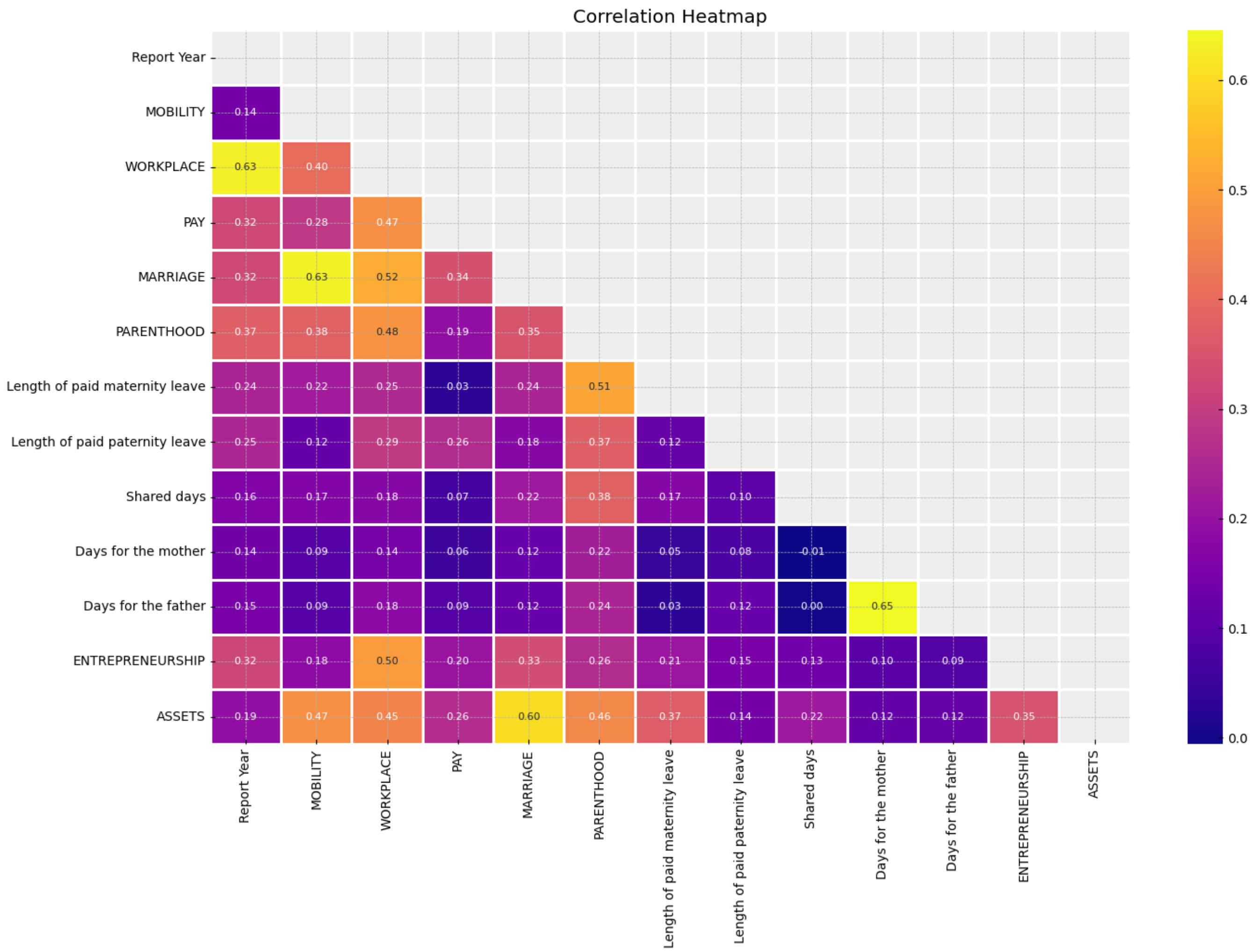
9.4 Correlacion Indice vs Paga

```
#plt.figure(dpi = 90,figsize= (10,10))
df1_ = df1.drop(['WBL INDEX','PAY'],axis=1)

plt.figure(figsize= (16,10))
mask = np.triu(np.ones_like(df1_.corr(),dtype = bool))
sns.heatmap(df1_.corr(),mask = mask, fmt = ".2f",annot=True,lw=1,cmap = 'plasma'
plt.yticks(rotation = 0)
plt.xticks(rotation = 90)
plt.title('Correlation Heatmap')
plt.show()
```



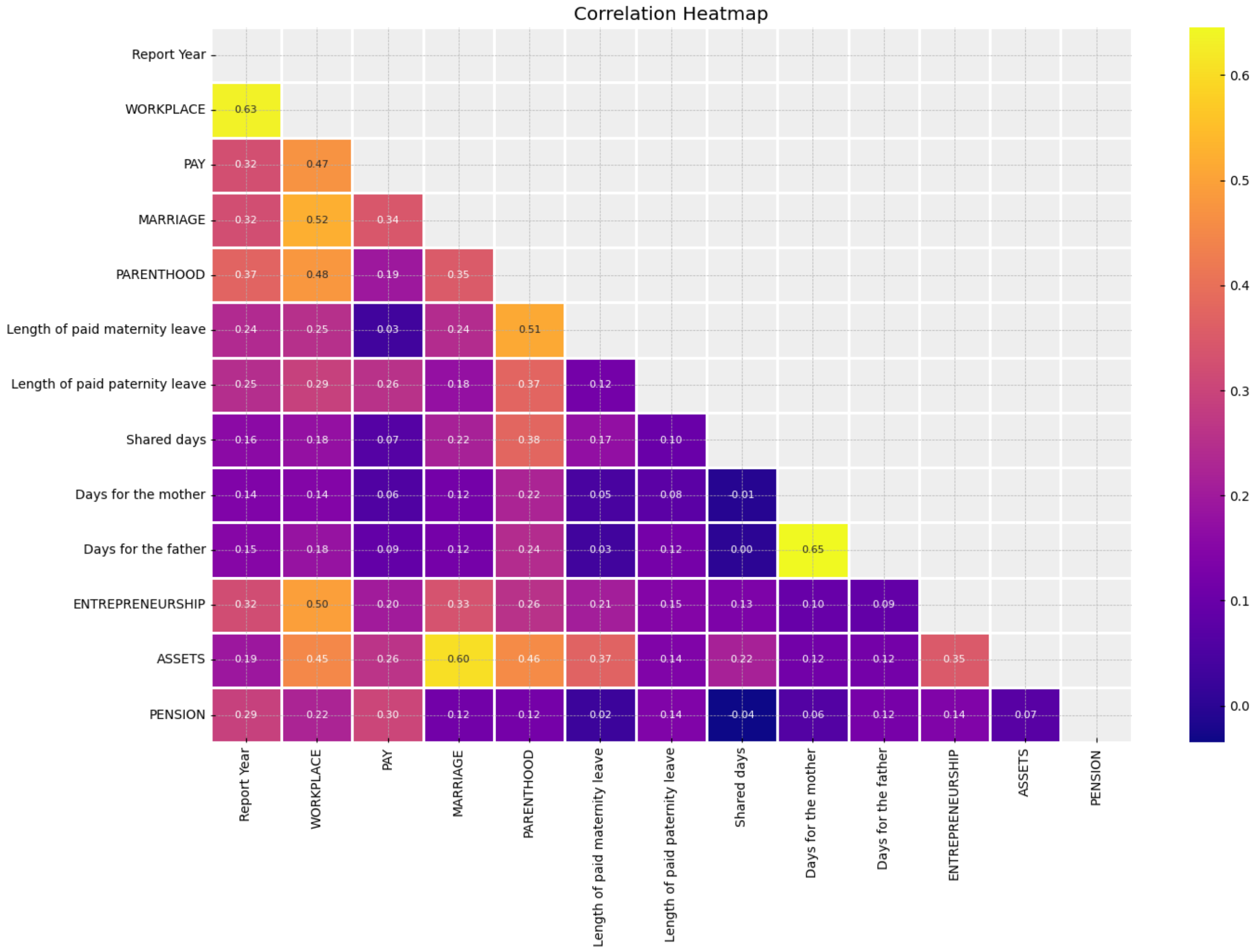
9.5 Correlacion Indice vs Pensiones

```
#plt.figure(dpi = 90,figsize= (10,10))
df1_ = df1.drop(['WBL INDEX','PENSION'],axis=1)

plt.figure(figsize= (16,10))
mask = np.triu(np.ones_like(df1_.corr(),dtype = bool))
sns.heatmap(df1_.corr(),mask = mask, fmt = ".2f",annot=True,lw=1,cmap = 'plasma'
plt.yticks(rotation = 0)
plt.xticks(rotation = 90)
plt.title('Correlation Heatmap')
plt.show()
```



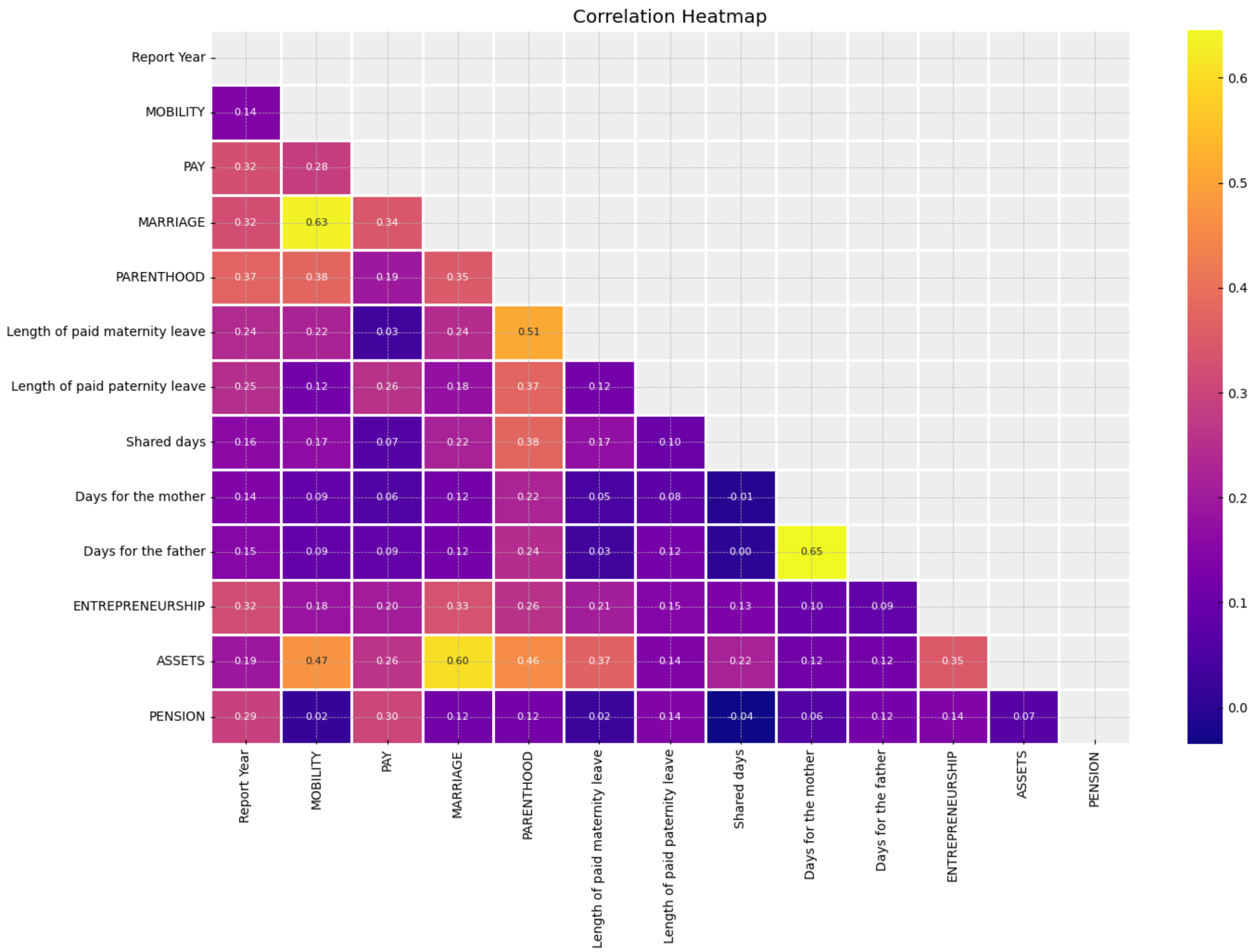
9.6 Correlacion Indice vs Mobilidad

```
#plt.figure(dpi = 90,figsize= (10,10))
df1_ = df1.drop(['WBL INDEX','MOBILITY'],axis=1)

plt.figure(figsize= (16,10))
mask = np.triu(np.ones_like(df1_.corr(),dtype = bool))
sns.heatmap(df1_.corr(),mask = mask, fmt = ".2f",annot=True,lw=1,cmap = 'plasma'
plt.yticks(rotation = 0)
plt.xticks(rotation = 90)
plt.title('Correlation Heatmap')
plt.show()
```



9.7 Correlacion Indice vs Lugar de trabajo

```
#plt.figure(dpi = 90,figsize= (10,10))
df1_ = df1.drop(['WBL INDEX','WORKPLACE'],axis=1)

plt.figure(figsize= (16,10))
mask = np.triu(np.ones_like(df1_.corr(),dtype = bool))
sns.heatmap(df1_.corr(),mask = mask, fmt = ".2f",annot=True,lw=1,cmap = 'plasma'
plt.yticks(rotation = 0)
plt.xticks(rotation = 90)
plt.title('Correlation Heatmap')
plt.show()
```

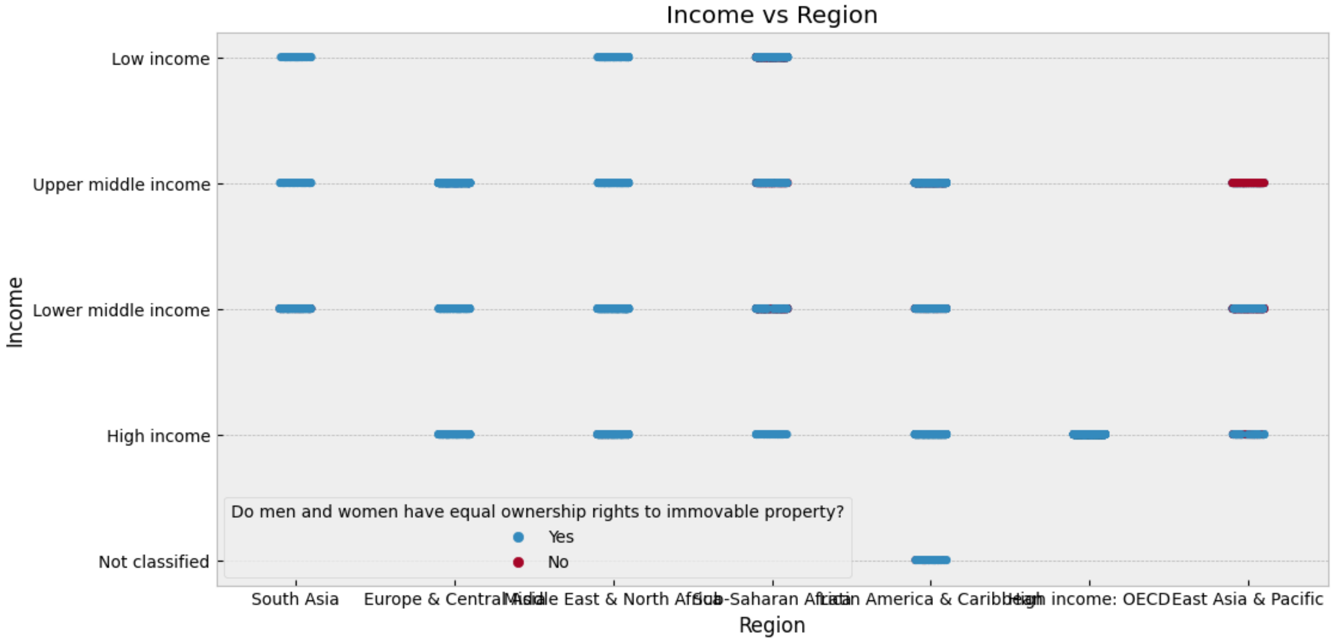


ANALISIS De los sucesivos analisis, podemos notar que las correlaciones entre variables con indices mayores a 0,6 se verifican que son pocas las variables que llegan arriba de esos valores. Notando que en definitiva, si existe un vinculo no se trata de un vinculo directo.

10 Evolucion de Derechos relaciones sobre ingreso y regiones

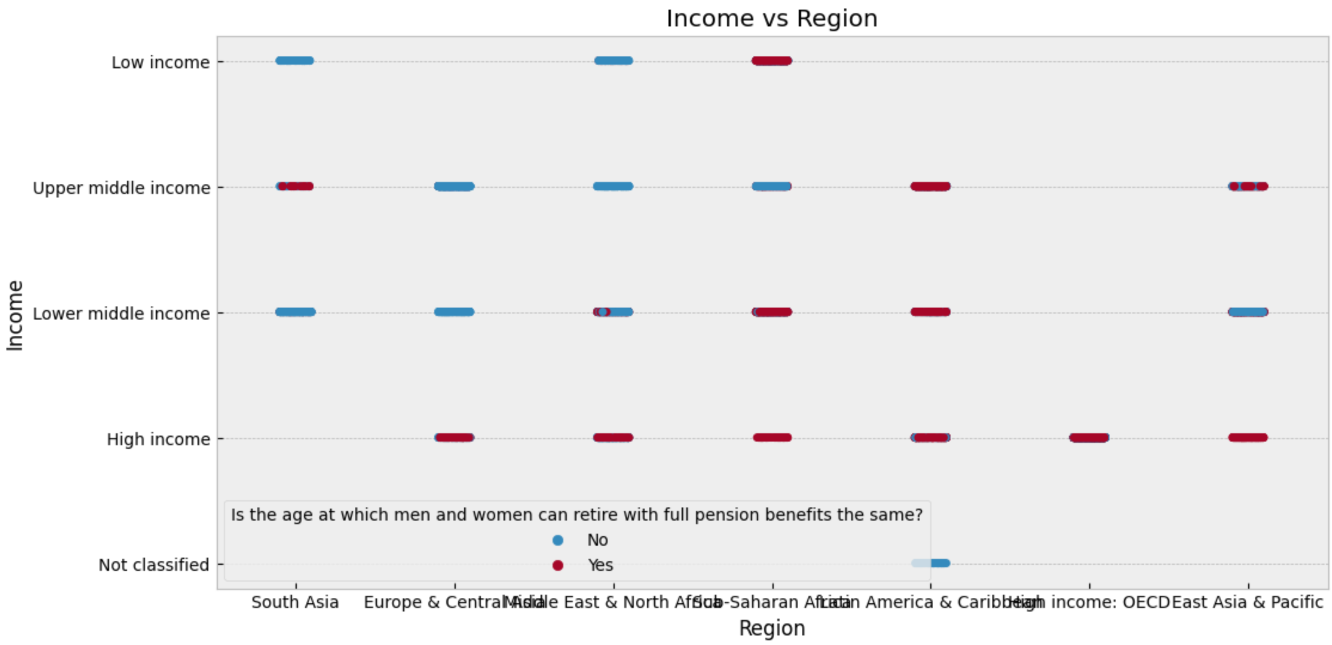
10.1 Derechos de la Propiedad relaciones sobre ingreso y regiones

```
plt.subplots(figsize=(12,6))
plt.title('Income vs Region')
ax = sns.stripplot(x="Region", y="Income Group", hue = "Do men and women have eq
plt.ylabel('Income')
plt.show()
```



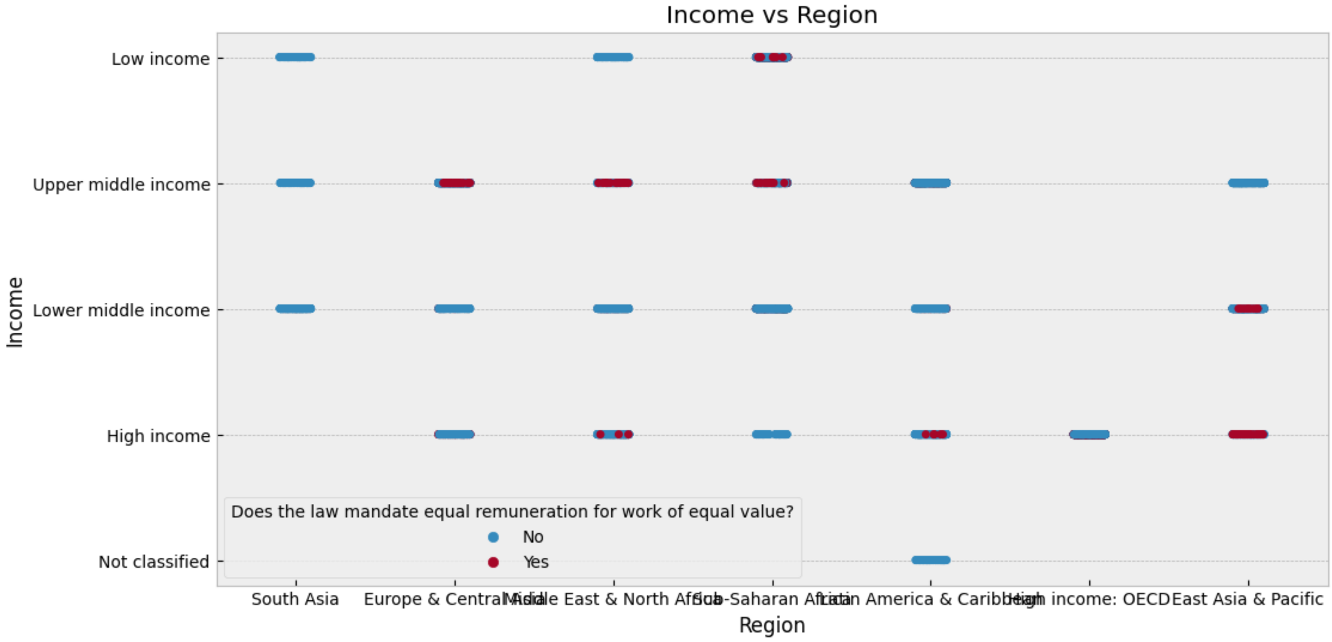
10.2 Seguridad social relaciones sobre ingreso y regiones

```
plt.subplots(figsize=(12,6))
plt.title('Income vs Region')
ax = sns.stripplot(x="Region", y="Income Group", hue = "Is the age at which men
plt.ylabel('Income')
plt.show()
```



10.3 Remuneracion y relaciones sobre ingreso y regiones

```
plt.subplots(figsize=(12,6))
plt.title('Income vs Region')
ax = sns.stripplot(x="Region", y="Income Group", hue = "Does the law mandate equ
plt.ylabel('Income')
plt.show()
```



ANALISIS En lo relativo a los Derechos de la propiedad, existe una relacion directa entre bienes e ingresos, pero es indistinto en lo referido a las zonas. Con respecto a los beneficios de pensiones no se notan en las clases sociales altas pero si en los sectores sociales medios y bajos, cuestion que llama bastante la atencion. En lo relativo a los ingresos pasa lo inverso.

11 Limpieza y filtrado de Datos

```
df1_1 = df1.copy()

df1_1 = df1_1.drop(columns=['Length of paid paternity leave', 'Days for the fath

df1_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10070 entries, 0 to 10069
Data columns (total 25 columns):
 #   Column
---  ------
 0   Economy
 1   Income Group
 2   Report Year
 3   WBL INDEX
 4   MOBILITY
 5   Can a woman choose where to live in the same way as a man?
 6   WORKPLACE
 7   Can a woman get a job in the same way as a man?
 8   Does the law prohibit discrimination in employment based on gender?
 9   Is there legislation on sexual harassment in employment?
 10  PAY
 11  Does the law mandate equal remuneration for work of equal value?
 12  MARRIAGE
 13  Is there legislation specifically addressing domestic violence?
 14  Can a woman obtain a judgment of divorce in the same way as a man?
 15  PARENTHOOD
 16  Length of paid maternity leave
 17  Shared days
 18  Days for the mother
 19  ENTREPRENEURSHIP
 20  Can a woman register a business in the same way as a man?
 21  ASSETS
 22  Do men and women have equal ownership rights to immovable property?
 23  PENSION
 24  Is the age at which men and women can retire with full pension benefit
dtypes: float64(1), int64(12), object(12)
memory usage: 1.9+ MB
```

11.1 Transformacion de variables

```
le_Economy = preprocessing.LabelEncoder()
df1_1['Economy']   = le_Economy.fit_transform(df1_1['Economy'] )
le_IncomeGroup = preprocessing.LabelEncoder()
df1_1['Income Group']  = le_IncomeGroup .fit_transform(df1_1['Income Group'] )
```

```
le_Canawomanchoosewheretoliveinthesamewayasaman = preprocessing.LabelEncoder()
df1_1['Can a woman choose where to live in the same way as a man?']  = le_Canawo
le_Canawomangetajobinthesamewayasaman = preprocessing.LabelEncoder()
df1_1['Can a woman get a job in the same way as a man?']  = le_Canawomangetajobi
le_Doestlawdiscriminationgender = preprocessing.LabelEncoder()
df1_1['Does the law prohibit discrimination in employment based on gender?']   =
le_Isexualharassmentemployment = preprocessing.LabelEncoder()
df1_1['Is there legislation on sexual harassment in employment?']   = le_Isexual
le_Doesequalremuneration = preprocessing.LabelEncoder()
df1_1['Does the law mandate equal remuneration for work of equal value?']   = le_
le_Candivorcesameasman = preprocessing.LabelEncoder()
df1_1['Can a woman obtain a judgment of divorce in the same way as a man?']   =
le_Canregisterbusinesssameasman = preprocessing.LabelEncoder()
df1_1['Can a woman register a business in the same way as a man?']   = le_Canreg
le_Doequalownershiprightsimmovableproperty = preprocessing.LabelEncoder()
df1_1['Do men and women have equal ownership rights to immovable property?']   =
le_Isageretirefullpensionsame  = preprocessing.LabelEncoder()
df1_1['Is the age at which men and women can retire with full pension benefits t
le_Idomesticviolence = preprocessing.LabelEncoder()
df1_1['Is there legislation specifically addressing domestic violence?']   = le_
df1_1
```

	Economy	Income Group	Report Year	WBL INDEX	MOBILITY	Can a woman choose where to live in the same way as a man?	WORKPLACE	Can a woman get a job in the same way as a man?	I dis ir
0	0	1	1971	26.250	25	0	25	1	
1	0	1	1972	26.250	25	0	25	1	
2	0	1	1973	26.250	25	0	25	1	
3	0	1	1974	26.250	25	0	25	1	
4	0	1	1975	26.250	25	0	25	1	
...	...	...	...	...	...	...	...	...	
10065	189	2	2019	86.875	100	1	100	1	
10066	189	2	2020	86.875	100	1	100	1	
10067	189	2	2021	86.875	100	1	100	1	
10068	189	2	2022	86.875	100	1	100	1	
10069	189	2	2023	86.875	100	1	100	1	

10070 rows × 25 columns

11.2 Media de variables

```
print('----------------------')
print('Media de cada variable')
print('----------------------')
df1_1.mean(axis=0)
```

```
----------------------
Media de cada variable
----------------------
Economy
94.500000
Income Group
```

```
1.778947
Report Year
1997.000000
WBL INDEX
59.725919
MOBILITY
82.142502
Can a woman choose where to live in the same way as a man?
0.678749
WORKPLACE
43.083416
Can a woman get a job in the same way as a man?
0.796624
Does the law prohibit discrimination in employment based on gender?
0.421946
Is there legislation on sexual harassment in employment?
0.263456
PAY
47.656405
Does the law mandate equal remuneration for work of equal value?
0.207944
MARRIAGE
61.813307
Is there legislation specifically addressing domestic violence?
0.273287
Can a woman obtain a judgment of divorce in the same way as a man?
0.690169
PARENTHOOD
34.975174
Length of paid maternity leave
85.440020
Shared days
38.561867
Days for the mother
5.271500
ENTREPRENEURSHIP
72.715988
Can a woman register a business in the same way as a man?
0.917180
ASSETS
73.769613
Do men and women have equal ownership rights to immovable property?
0.790367
PENSION
61.650943
Is the age at which men and women can retire with full pension benefits
the same?       0.504171
dtype: float64
```

11.3 Varianza

```
print('-------------------------')
print('Varianza de cada variable')
print('-------------------------')
df1_1.var(axis=0)
```

```
-------------------------
Varianza de cada variable
-------------------------
Economy
3008.548764
Income Group
2.372424
Report Year
234.023240
WBL INDEX
346.868461
MOBILITY
662.245557
Can a woman choose where to live in the same way as a man?
0.218071
WORKPLACE
1135.615248
Can a woman get a job in the same way as a man?
0.162031
Does the law prohibit discrimination in employment based on gender?
0.243932
Is there legislation on sexual harassment in employment?
0.194066
PAY
978.095259
Does the law mandate equal remuneration for work of equal value?
0.164720
MARRIAGE
893.523589
Is there legislation specifically addressing domestic violence?
0.198621
Can a woman obtain a judgment of divorce in the same way as a man?
0.213857
PARENTHOOD
938.578190
Length of paid maternity leave
3857.636237
Shared days
25577.474323
Days for the mother
1643.128585
ENTREPRENEURSHIP
450.761518
Can a woman register a business in the same way as a man?
0.075969
ASSETS
852.667147
Do men and women have equal ownership rights to immovable property?
0.165703
```

```
PENSION
839.574741
Is the age at which men and women can retire with full pension benefits
the same?        0.250007
dtype: float64
```

```
df1_1.shape
```

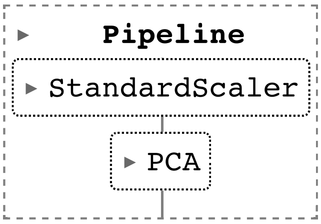
```
(10070, 25)
```

12 Analisis de componentes

```
pca_pipe = make_pipeline(StandardScaler(), PCA())
pca_pipe.fit(df1_1)
```

```
modelo_pca = pca_pipe.named_steps['pca']
```

```
pca_pipe.fit(df1_1)
```




```
pd.DataFrame(
    data    = modelo_pca.components_,
    columns = df1_1.columns,
    index   = ['PC1', 'PC2', 'PC3', 'PC4', 'PC5', 'PC6',
               'PC7', 'PC8', 'PC9', 'PC10', 'PC11', 'PC12',
               'PC13', 'PC14', 'PC15', 'PC16', 'PC17', 'PC18',
               'PC19', 'PC20', 'PC21', 'PC22', 'PC23', 'PC24',
               'PC25']
)
```

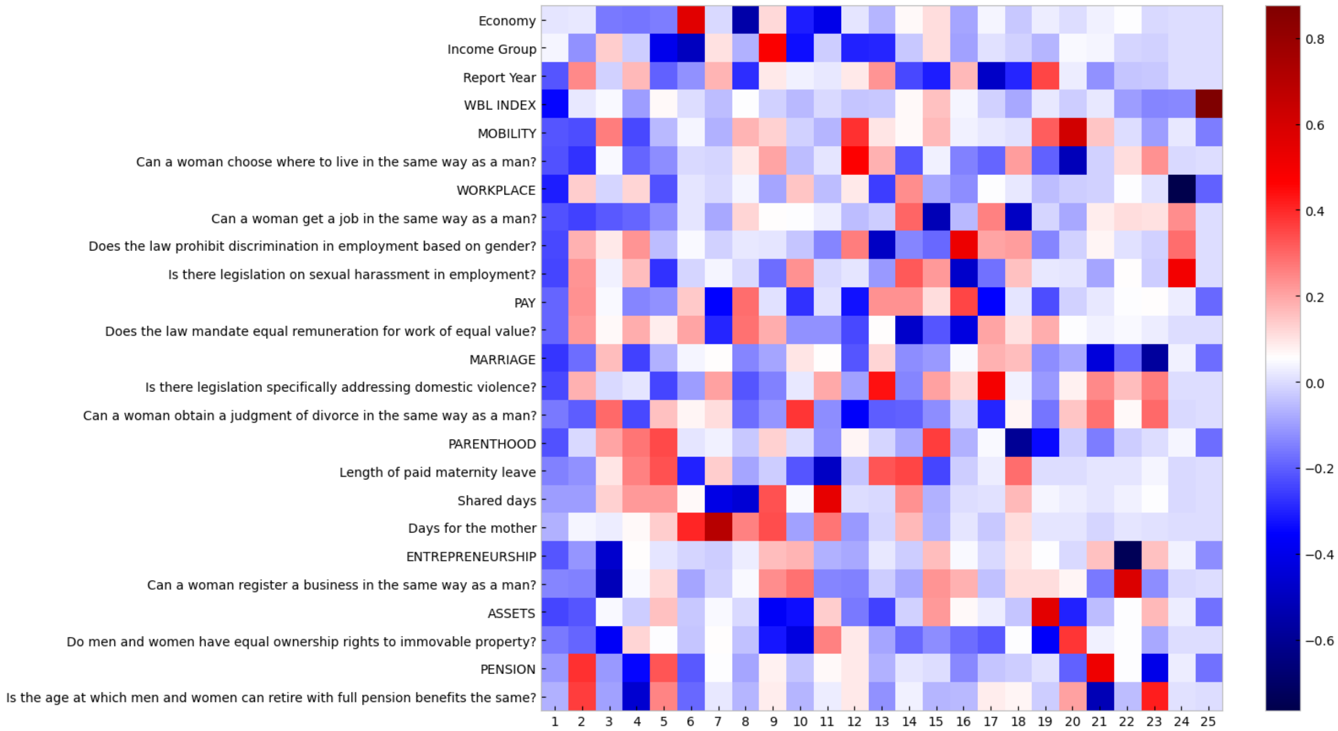
Can a woman

	Economy	Income Group	Report Year	WBL INDEX	MOBILITY	choose where to live in the same way as a man?	WORKPLA
PC1	1.451214e-02	3.733748e-02	-2.188192e-01	-0.345031	-0.214661	-2.229140e-01	-0.3066
PC2	1.903800e-02	-1.237730e-01	2.450320e-01	0.019487	-0.230886	-2.750894e-01	0.1356
PC3	-1.599607e-01	1.361695e-01	-1.559993e-02	0.045908	0.260799	4.620882e-02	-0.0107
PC4	-1.668838e-01	-2.497264e-02	1.691961e-01	-0.098742	-0.233758	-1.896798e-01	0.1250
PC5	-1.533083e-01	-3.953108e-01	-1.947381e-01	0.066838	-0.057074	-1.252136e-01	-0.2254
PC6	5.580485e-01	-4.972060e-01	-1.189620e-01	0.001748	0.039045	-2.657372e-03	0.0130
PC7	-2.311904e-03	1.038695e-01	1.739394e-01	-0.050920	-0.066541	-1.065946e-02	-0.0027
PC8	-5.510097e-01	-6.642479e-02	-2.779302e-01	0.051347	0.176561	8.918808e-02	0.0375
PC9	1.143118e-01	4.773726e-01	8.757092e-02	-0.020588	0.128491	1.981825e-01	-0.0893
PC10	-3.071264e-01	-3.315598e-01	3.543219e-02	-0.056429	-0.017409	-4.954509e-02	0.1470
PC11	-4.058034e-01	-2.461265e-02	1.809413e-02	-0.005433	-0.065844	1.356335e-02	-0.0476
PC12	1.230842e-02	-3.002485e-01	9.053914e-02	-0.034646	0.388055	4.675935e-01	0.0894
PC13	-6.417503e-02	-2.912361e-01	2.254391e-01	-0.029832	0.100189	1.800119e-01	-0.2568
PC14	6.269697e-02	-3.414207e-02	-2.372262e-01	0.067065	0.061881	-2.171977e-01	0.2408
PC15	1.070948e-01	1.113387e-01	-3.057093e-01	0.152083	0.166186	3.078336e-02	-0.0817
PC16	-9.155153e-02	-9.766121e-02	1.667567e-01	0.041635	0.031930	-1.460543e-01	-0.1241

12.1 Ejes

-3.183113e- -1.826534e- -2.910392e- 2.155841e-

```
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(12, 10))
componentes = modelo_pca.components_
plt.imshow(componentes.T, cmap='seismic', aspect='auto')
plt.yticks(range(len(df1_1.columns)), df1_1.columns)
plt.xticks(range(len(df1_1.columns)), np.arange(modelo_pca.n_components_) + 1)
plt.grid(False)
plt.colorbar();
```



ANALISIS Por la alta cantidad de variables se hace confuso, igualmente se nota diferenciados los ejes.

```
modelo_pca.n_components_
```

    25

```
np.arange(len(df1_1.columns)) + 1
```

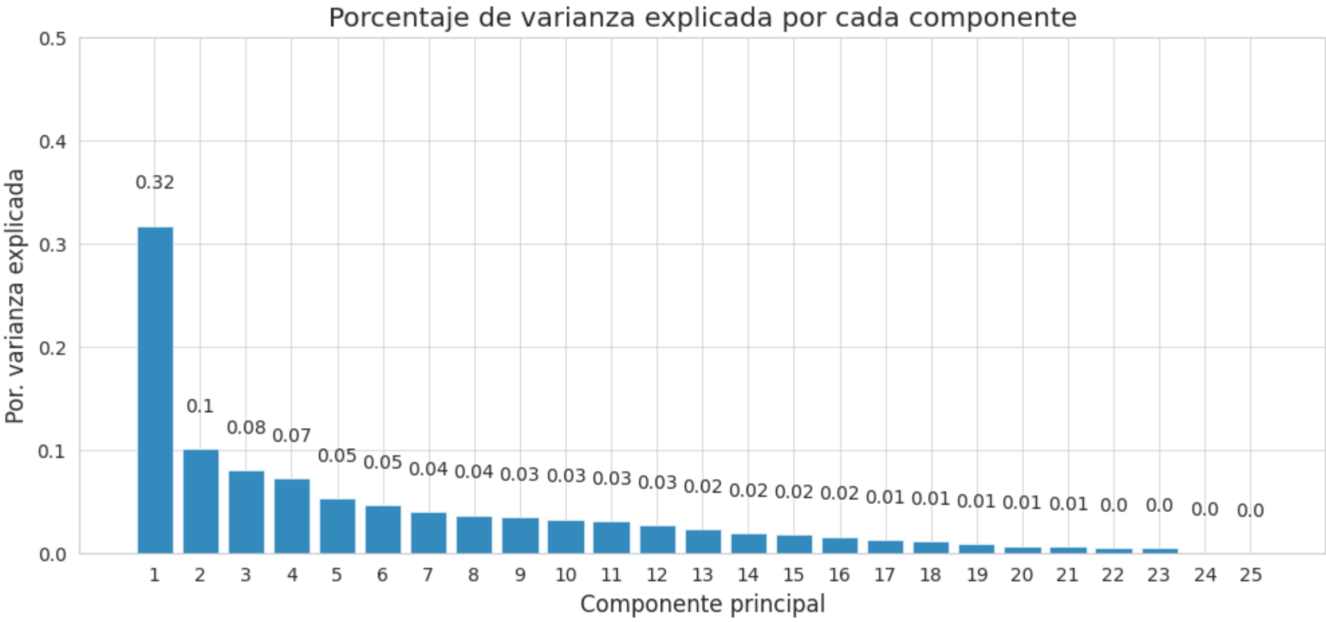
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25])
```

```
modelo_pca.explained_variance_ratio_
```

```
array([3.16970949e-01, 1.01156300e-01, 8.00370605e-02, 7.19089389e-02,
       5.27860474e-02, 4.63102029e-02, 4.00254850e-02, 3.66061254e-02,
       3.41505847e-02, 3.25554005e-02, 3.02999519e-02, 2.69270758e-02,
       2.27108463e-02, 1.87816129e-02, 1.73438225e-02, 1.58659981e-02,
       1.30278880e-02, 1.15494905e-02, 8.68109548e-03, 6.72044735e-03,
       5.75262198e-03, 4.99194099e-03, 4.56362661e-03, 2.76487143e-04,
       2.30952718e-33])
```

```
print('----------------------------------------------------')
print('Porcentaje de varianza explicada por cada componente')
print('----------------------------------------------------')
print(modelo_pca.explained_variance_ratio_)
import seaborn as sns
sns.set_style("whitegrid")

fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(12, 5))
ax.bar(
    x      = np.arange(modelo_pca.n_components_) + 1,
    height = modelo_pca.explained_variance_ratio_
)

for x, y in zip(np.arange(len(df1_1.columns)) + 1, modelo_pca.explained_variance_
    label = round(y, 2)
    ax.annotate(
        label,
        (x,y),
        textcoords="offset points",
        xytext=(0,20),
        ha='center'
    )

ax.set_xticks(np.arange(modelo_pca.n_components_) + 1)
ax.set_ylim(0, 0.5)
ax.set_title('Porcentaje de varianza explicada por cada componente')
ax.set_xlabel('Componente principal')
ax.set_ylabel('Por. varianza explicada');
```

```
----------------------------------------------------
Porcentaje de varianza explicada por cada componente
----------------------------------------------------
[3.16970949e-01 1.01156300e-01 8.00370605e-02 7.19089389e-02
 5.27860474e-02 4.63102029e-02 4.00254850e-02 3.66061254e-02
 3.41505847e-02 3.25554005e-02 3.02999519e-02 2.69270758e-02
 2.27108463e-02 1.87816129e-02 1.73438225e-02 1.58659981e-02
 1.30278880e-02 1.15494905e-02 8.68109548e-03 6.72044735e-03
 5.75262198e-03 4.99194099e-03 4.56362661e-03 2.76487143e-04
 2.30952718e-33]
```



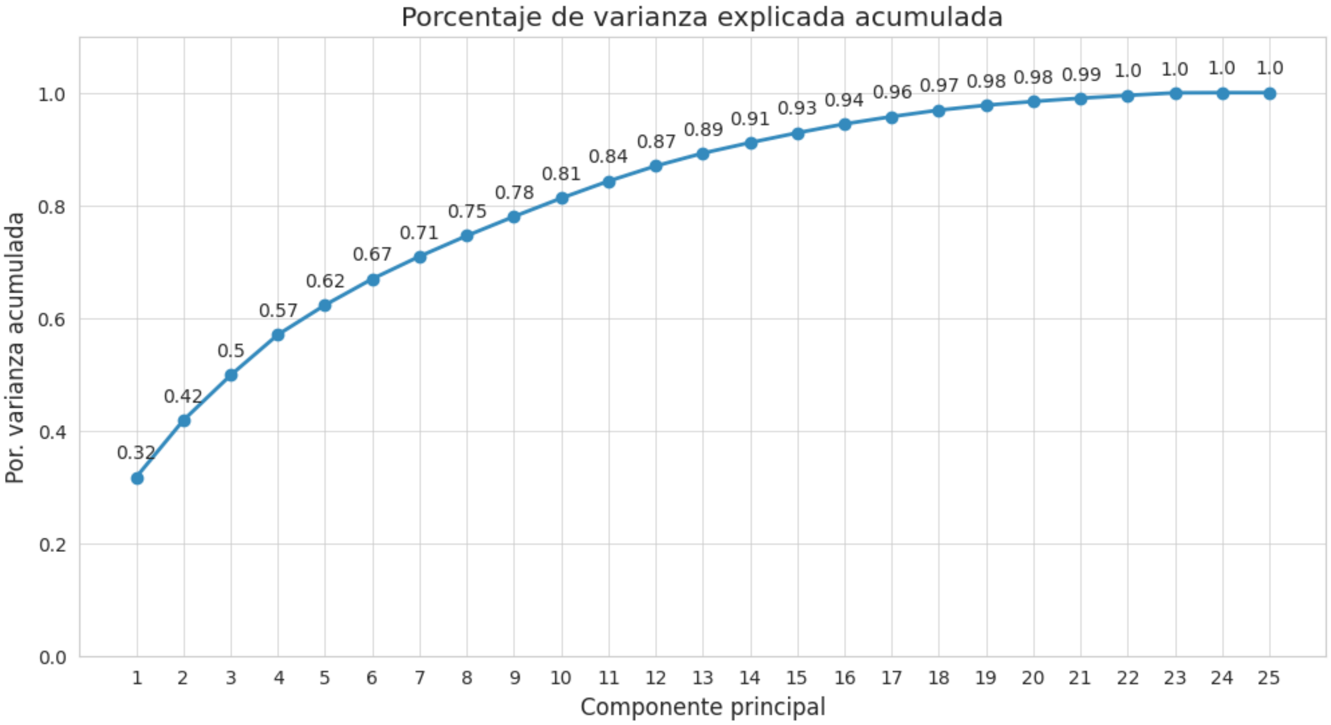

```python
prop_varianza_acum = modelo_pca.explained_variance_ratio_.cumsum()
prop_varianza_acum
```

```
array([0.31697095, 0.41812725, 0.49816431, 0.57007325, 0.6228593 ,
       0.6691695 , 0.70919498, 0.74580111, 0.77995169, 0.81250709,
       0.84280705, 0.86973412, 0.89244497, 0.91122658, 0.9285704 ,
       0.9444364 , 0.95746429, 0.96901378, 0.97769488, 0.98441532,
       0.99016795, 0.99515989, 0.99972351, 1.        , 1.        ])
```

```python
prop_varianza_acum = modelo_pca.explained_variance_ratio_.cumsum()
print('----------------------------------------')
print('Porcentaje de varianza explicada acumulada')
print('----------------------------------------')
print(prop_varianza_acum)
```

```
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(12, 6))
ax.plot(
    np.arange(len(df1_1.columns)) + 1,
    prop_varianza_acum,
    marker = 'o'
)

for x, y in zip(np.arange(len(df1_1.columns)) + 1, prop_varianza_acum):
    label = round(y, 2)
    ax.annotate(
        label,
        (x,y),
        textcoords="offset points",
        xytext=(0,10),
        ha='center'
    )

ax.set_ylim(0, 1.1)
ax.set_xticks(np.arange(modelo_pca.n_components_) + 1)
ax.set_title('Porcentaje de varianza explicada acumulada')
ax.set_xlabel('Componente principal')
ax.set_ylabel('Por. varianza acumulada');
```

```
------------------------------------------
Porcentaje de varianza explicada acumulada
------------------------------------------
[0.31697095 0.41812725 0.49816431 0.57007325 0.6228593  0.6691695
 0.70919498 0.74580111 0.77995169 0.81250709 0.84280705 0.86973412
 0.89244497 0.91122658 0.9285704  0.9444364  0.95746429 0.96901378
 0.97769488 0.98441532 0.99016795 0.99515989 0.99972351 1.
 1.        ]
```



13 Regresion Linear

```
from sklearn import datasets
from sklearn.linear_model import LinearRegression
import numpy as np
```

```
X = df1[['MOBILITY', 'PENSION', 'MARRIAGE','WORKPLACE']]
y = df1['WBL INDEX']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
regressor.coef_
```

```
array([0.1804499 , 0.1587452 , 0.19964468, 0.26426854])
```

```
regressor.intercept_
```

```
11.331810502380968
```

```
coeff_df = pd.DataFrame(regressor.coef_, X.columns, columns=['Coefficient'])
coeff_df
```

	Coefficient
MOBILITY	0.180450
PENSION	0.158745
MARRIAGE	0.199645
WORKPLACE	0.264269

```
y_pred = regressor.predict(X_test)
y_pred
```

```
array([36.30161629, 55.92371846, 55.92371846, ..., 66.49906191,
       63.83671495, 51.95508845])
```

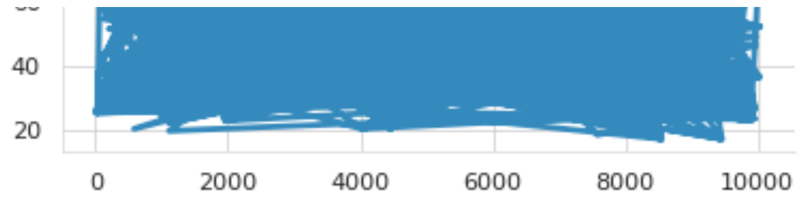
```
y_test

    1121    22.500
    318     55.625
    9133    50.000
    1149    61.250
    33      26.250
             ...
    1362    76.875
    1437    36.875
    9696    70.625
    8601    63.750
    56      59.375
    Name: WBL INDEX, Length: 3021, dtype: float64
```

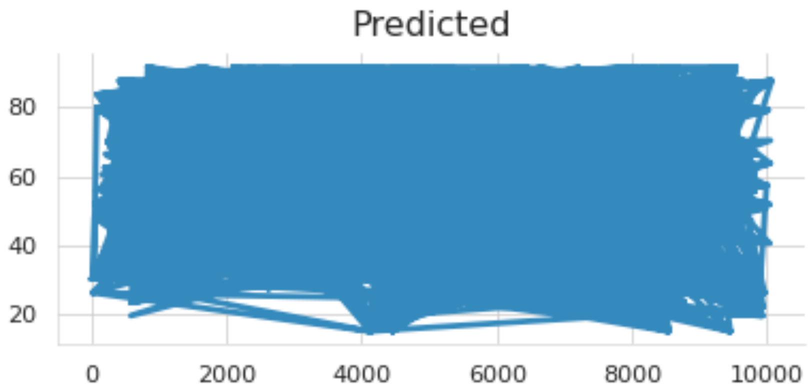
```
df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
df['Sesgo']=df.Actual -df.Predicted
df['Error_porc']=((df.Actual -df.Predicted)/df.Actual) *100
df
```

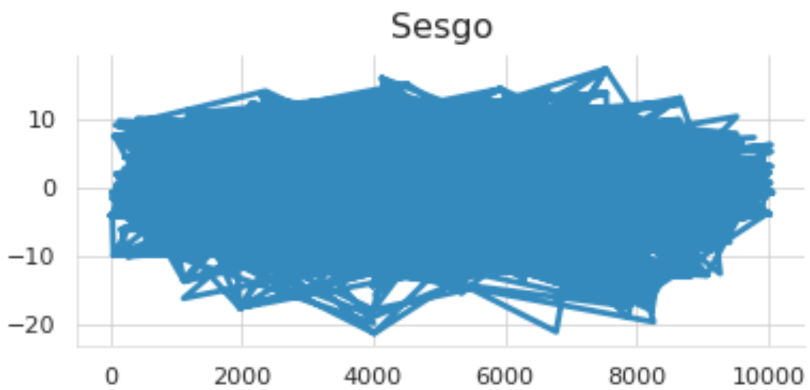
	Actual	Predicted	Sesgo	Error_porc
1121	22.500	36.301616	-13.801616	-61.340517
318	55.625	55.923718	-0.298718	-0.537022
9133	50.000	55.923718	-5.923718	-11.847437
1149	61.250	63.318361	-2.068361	-3.376916
33	26.250	30.411295	-4.161295	-15.852553
...	...	...	...	...
1362	76.875	79.145608	-2.270608	-2.953636
1437	36.875	36.819970	0.055030	0.149233
9696	70.625	66.499062	4.125938	5.842036
8601	63.750	63.836715	-0.086715	-0.136023
56	59.375	51.955088	7.419912	12.496693

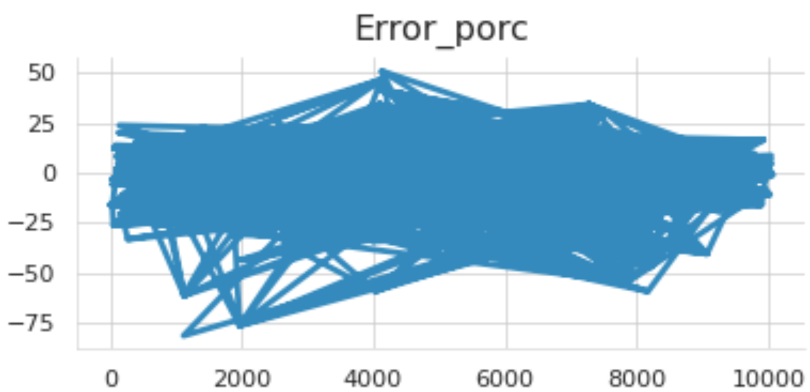
3021 rows × 4 columns

Values

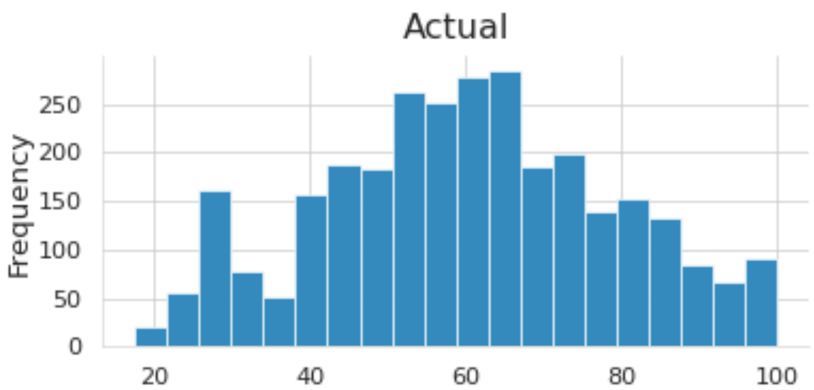


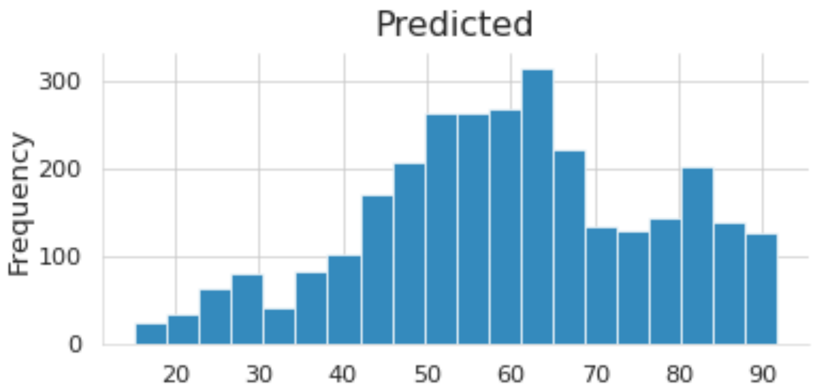


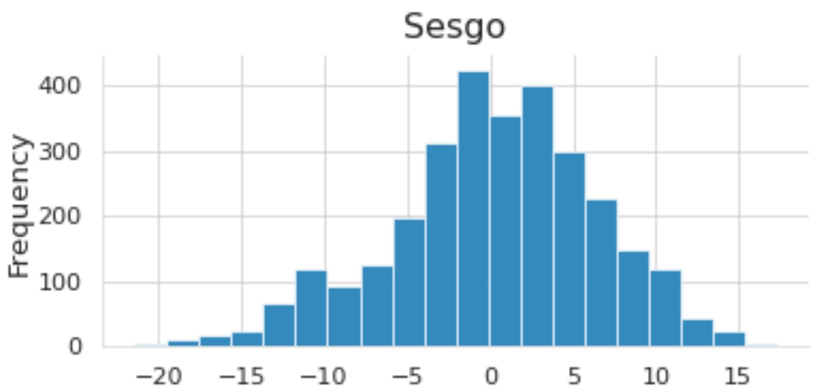


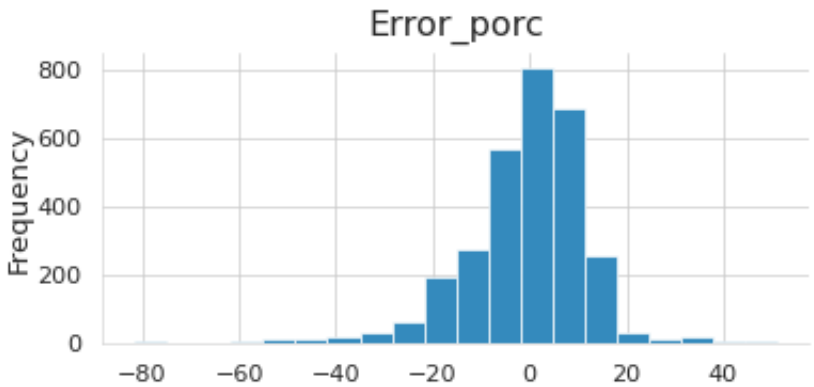


Distributions

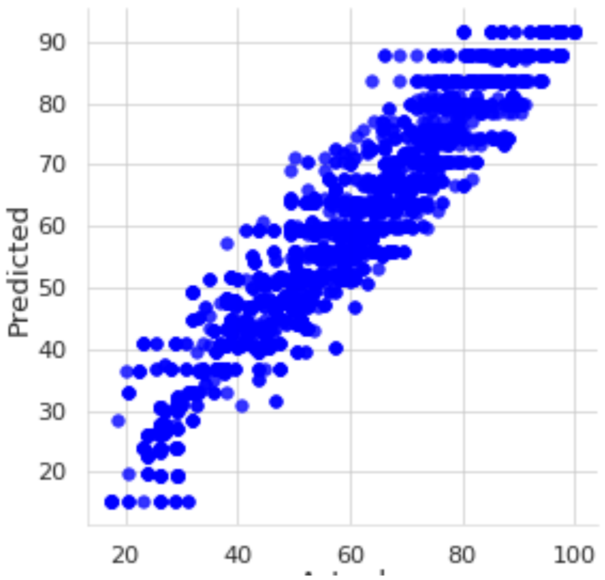


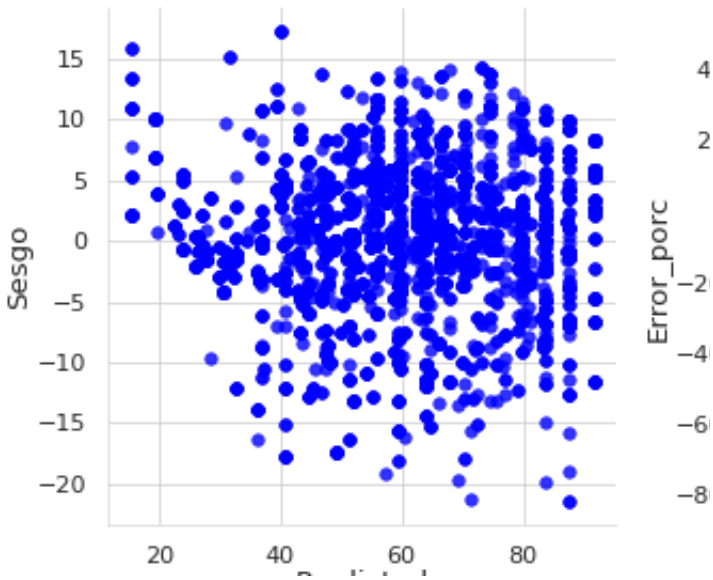






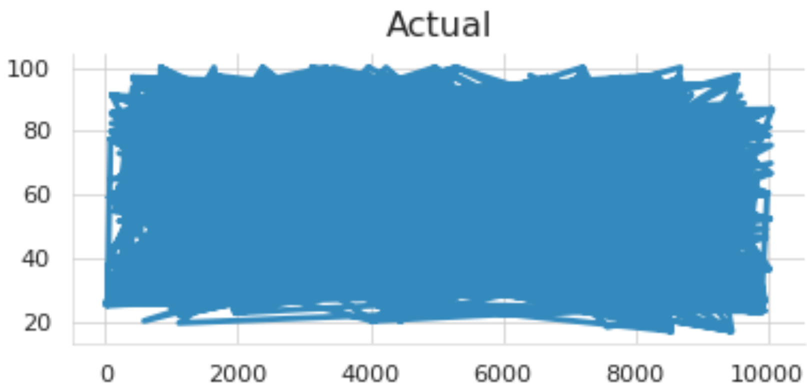
2-d distributions

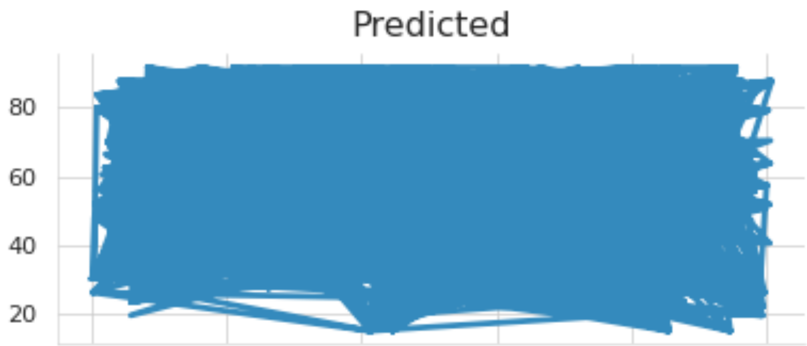




Actual Predicted

Values





ANALISIS En el modelo de regresion linear, en lo relativo a matrimonio, pension, matrimonio y lugar de trabajo, en los modelos actuales la prediccion demuestra un avance pero en el modelo predicho no se nota de manera uniforme ese crecimiento. La nota en particular, por el tipo de variable es alto el sesgo.

SEGUNDA PARTE PROYECTO FINAL PARTE 2

```
import numpy as np
import pandas as pd
import scipy as sp
from prettytable import PrettyTable


import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px

mpl.style.use('bmh')


from sklearn.decomposition import PCA
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sm
from sklearn.utils import resample
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import GridSearchCV
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import roc_curve, roc_auc_score
import warnings
warnings.filterwarnings('ignore')
```

```
df1= pd.read_excel("/content/drive/MyDrive/Datasetwl (1).xlsx")
df1.head()
```

	Economy	Economy Code	ISO Code	Region	Income Group	Report Year	WBL INDEX	MOBILITY	Can a woman choose where to live in the same way as a man?
0	Afghanistan	AFG	AFG	South Asia	Low income	1971	26.25	25	No
1	Afghanistan	AFG	AFG	South Asia	Low income	1972	26.25	25	No
2	Afghanistan	AFG	AFG	South Asia	Low income	1973	26.25	25	No
3	Afghanistan	AFG	AFG	South Asia	Low income	1974	26.25	25	No
4	Afghanistan	AFG	AFG	South Asia	Low income	1975	26.25	25	No

5 rows × 55 columns

```
df1_1 = df1.copy()

df1_1 = df1_1.drop(columns=['Length of paid paternity leave', 'Days for the fath

df1_1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10070 entries, 0 to 10069
Data columns (total 25 columns):
 #   Column
---  ------
 0   Economy
 1   Income Group
 2   Report Year
 3   WBL INDEX
 4   MOBILITY
 5   Can a woman choose where to live in the same way as a man?
 6   WORKPLACE
 7   Can a woman get a job in the same way as a man?
 8   Does the law prohibit discrimination in employment based on gender?
 9   Is there legislation on sexual harassment in employment?
 10  PAY
 11  Does the law mandate equal remuneration for work of equal value?
 12  MARRIAGE
 13  Is there legislation specifically addressing domestic violence?
 14  Can a woman obtain a judgment of divorce in the same way as a man?
 15  PARENTHOOD
 16  Length of paid maternity leave
 17  Shared days
 18  Days for the mother
 19  ENTREPRENEURSHIP
 20  Can a woman register a business in the same way as a man?
 21  ASSETS
 22  Do men and women have equal ownership rights to immovable property?
 23  PENSION
 24  Is the age at which men and women can retire with full pension benefi
dtypes: float64(1), int64(12), object(12)
memory usage: 1.9+ MB
```

Transformacion de variables

```
le_Economy = preprocessing.LabelEncoder()
df1_1['Economy']   = le_Economy.fit_transform(df1_1['Economy'] )
le_IncomeGroup = preprocessing.LabelEncoder()
df1_1['Income Group']  = le_IncomeGroup .fit_transform(df1_1['Income Group'] )
le_Canawomanchoosewheretoliveinthesamewayasaman = preprocessing.LabelEncoder()
df1_1['Can a woman choose where to live in the same way as a man?']  = le_Canawo
```

```
le_Canawomangetajobinthesamewayasaman = preprocessing.LabelEncoder()
df1_1['Can a woman get a job in the same way as a man?']  = le_Canawomangetajobi
le_Doestlawdiscriminationgender = preprocessing.LabelEncoder()
df1_1['Does the law prohibit discrimination in employment based on gender?']   =
le_Isexualharassmentemployment = preprocessing.LabelEncoder()
df1_1['Is there legislation on sexual harassment in employment?']   = le_Isexual
le_Doesequalremuneration = preprocessing.LabelEncoder()
df1_1['Does the law mandate equal remuneration for work of equal value?']   = le
le_Candivorcesameasman = preprocessing.LabelEncoder()
df1_1['Can a woman obtain a judgment of divorce in the same way as a man?']   =
le_Canregisterbusinesssameasman = preprocessing.LabelEncoder()
df1_1['Can a woman register a business in the same way as a man?']   = le_Canreg
le_Doequalownershiprightsimmovableproperty = preprocessing.LabelEncoder()
df1_1['Do men and women have equal ownership rights to immovable property?']   =
le_Isageretirefullpensionsame  = preprocessing.LabelEncoder()
df1_1['Is the age at which men and women can retire with full pension benefits t
le_Idomesticviolence = preprocessing.LabelEncoder()
df1_1['Is there legislation specifically addressing domestic violence?']   = le_
df1_1
```

	Economy	Income Group	Report Year	WBL INDEX	MOBILITY	Can a woman choose where to live in the same way as a man?	WORKPLACE	Can a woman get a job in the same way as a man?	di i
0	0	1	1971	26.250	25	0	25	1	
1	0	1	1972	26.250	25	0	25	1	
2	0	1	1973	26.250	25	0	25	1	
3	0	1	1974	26.250	25	0	25	1	
4	0	1	1975	26.250	25	0	25	1	
...	...	...	...	...	...	...	...	...	
10065	189	2	2019	86.875	100	1	100	1	
10066	189	2	2020	86.875	100	1	100	1	
10067	189	2	2021	86.875	100	1	100	1	
10068	189	2	2022	86.875	100	1	100	1	
10069	189	2	2023	86.875	100	1	100	1	

10070 rows × 25 columns

14 Defino variables

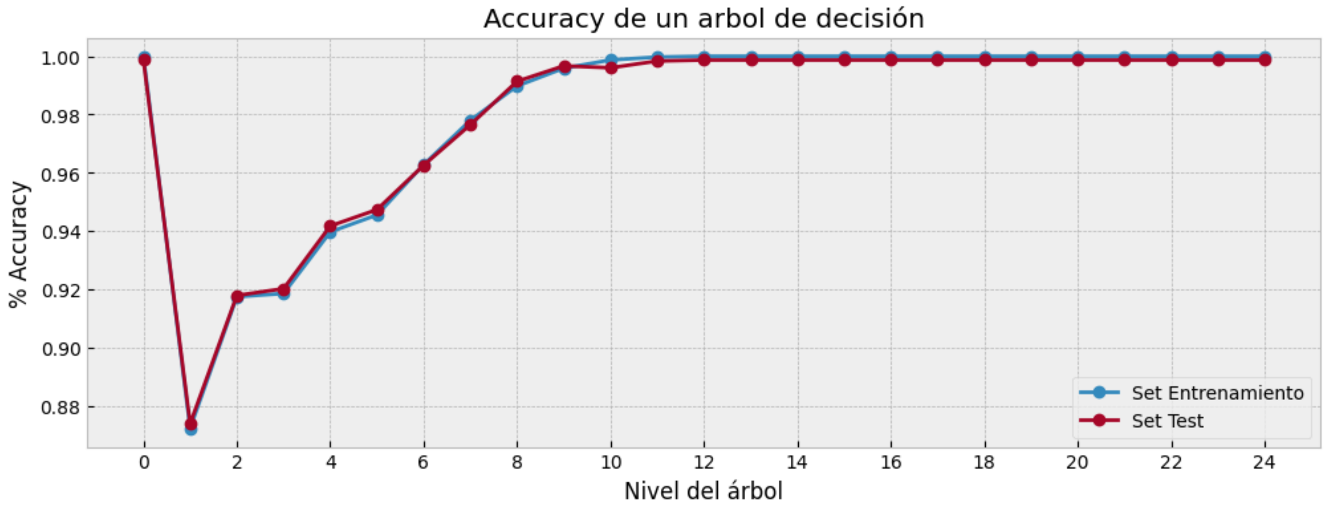
```
X = df1_1.drop("Do men and women have equal ownership rights to immovable proper
y = df1_1 ['Do men and women have equal ownership rights to immovable property?'
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random
```

15 Arbol de Decision

```
A_train = []
A_test = []
for i in range(0,25):

    if i == 0:
        i = None


    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, ra
    arbol_de_decision = DecisionTreeClassifier(max_depth=i, random_state = 11)
    arbol_de_decision.fit(X_train,y_train)
    y_train_pred = arbol_de_decision.predict(X_train)
    y_test_pred = arbol_de_decision.predict(X_test)
    train_accuracy = accuracy_score(y_train, y_train_pred)
    test_accuracy = accuracy_score(y_test, y_test_pred)


    A_train.append(train_accuracy)
    A_test.append(test_accuracy)
```

```
fig, ax = plt.subplots(figsize=(12, 4))

ax.plot(A_train,marker='o', label='Set Entrenamiento')
ax.plot(A_test,marker='o', label='Set Test', color='C1')

ax.set_xlabel('Nivel del árbol')
ax.set_xticks(range(0,26,2))
ax.set_ylabel('% Accuracy')
#ax.set_ylim(0, 1.1)
ax.set_title('Accuracy de un arbol de decisión')
ax.legend()
```

```
<matplotlib.legend.Legend at 0x7e92c60375b0>
```



```
arbol_de_decision = DecisionTreeClassifier(max_depth=4, random_state = 42)

arbol_de_decision.fit(X_train,y_train)

y_train_pred = arbol_de_decision.predict(X_train)

y_test_pred = arbol_de_decision.predict(X_test)

train_accuracy = accuracy_score(y_train, y_train_pred)
test_accuracy = accuracy_score(y_test, y_test_pred)
print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy,4))
#print('% de aciertos sobre el set de entrenamiento:', train_accuracy)
print('% de aciertos sobre el set de evaluación:',round(test_accuracy,3))
```

```
% de aciertos sobre el set de entrenamiento: 0.9397
% de aciertos sobre el set de evaluación: 0.942
```

```
train_accuracy_10_1_1 = accuracy_score(y_train, y_train_pred)

test_accuracy_10_1_1 = accuracy_score(y_test, y_test_pred)

print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_10_1_
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_10_1_1,3))
```

```
% de aciertos sobre el set de entrenamiento: 0.94
% de aciertos sobre el set de evaluación: 0.942
```

```
confusion_matrix(y_test, y_test_pred)
```

```
array([[ 565,   65],
       [ 111, 2280]])
```

```
precision_10_1_1 = round(precision_score(y_test, y_test_pred),5)
precision_10_1_1
```

```
0.97228
```

```
recall_10_1_1 = round(recall_score(y_test, y_test_pred),5)
recall_10_1_1
```

```
0.95358
```

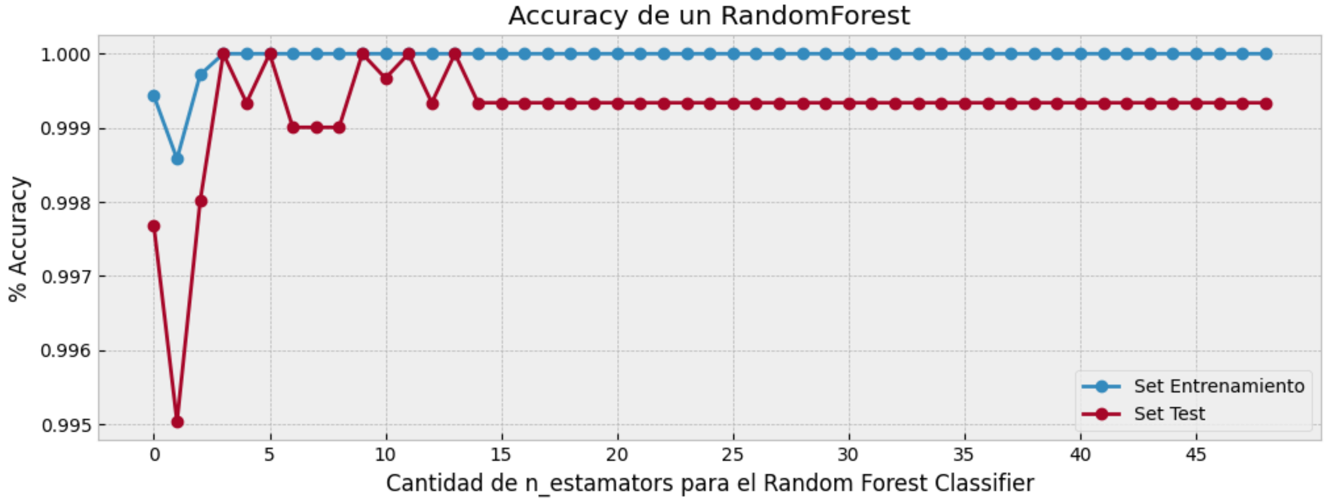
```
y_score1 = arbol_de_decision.predict_proba(X_test)[:,1]

false_positive_rate1, true_positive_rate1, threshold1 = roc_curve(y_test, y_scor
roc_10_1_1 = round(roc_auc_score(y_test, y_score1),5)
print('roc_auc_score for DecisionTree: ', roc_10_1_1)

plt.subplots(1, figsize=(8,6))
plt.title('Receiver Operating Characteristic - DecisionTree')
plt.plot(false_positive_rate1, true_positive_rate1, marker = 'o')
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

```
roc_auc_score for DecisionTree:  0.97201
```

16 Random Forest

```
A_train = []
A_test = []

for i in range(1,50):

    #Creamos el modelo
    random_forest_model = RandomForestClassifier(n_estimators = i, random_state
    #Entrenamos el modelo
    random_forest_model.fit(X_train,y_train)
    #Prediccion en Train y Test
    y_train_pred = random_forest_model.predict(X_train)
    y_test_pred = random_forest_model.predict(X_test)
    #Calculo el accuracy en Train y Test
    train_accuracy = accuracy_score(y_train, y_train_pred)
    test_accuracy = accuracy_score(y_test, y_test_pred)

    # Acumulo los resultados de Accuracy para cada corrida:

    A_train.append(train_accuracy)
    A_test.append(test_accuracy)

fig, ax = plt.subplots(figsize=(12, 4))

ax.plot(A_train,marker='o', label='Set Entrenamiento')
ax.plot(A_test,marker='o', label='Set Test', color='C1')

ax.set_xlabel('Cantidad de n_estamators para el Random Forest Classifier')
ax.set_xticks(range(0,50,5))
ax.set_ylabel('% Accuracy')
#ax.set_ylim(0, 1.1)
ax.set_title('Accuracy de un RandomForest')
ax.legend()
```

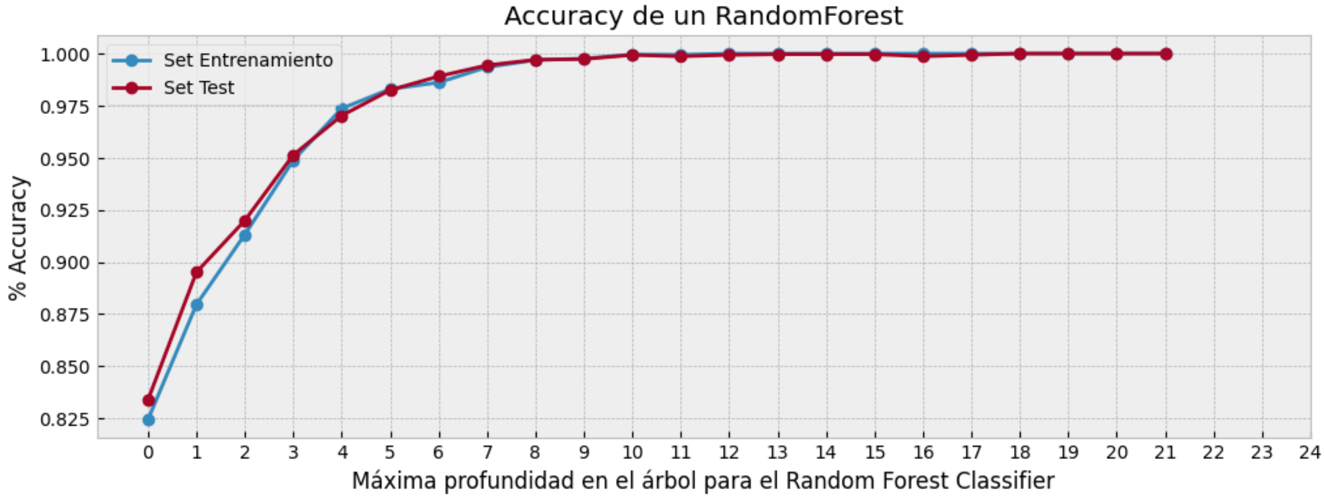
```
<matplotlib.legend.Legend at 0x7e92c5235540>
```




```
A_train = []
A_test = []

for i in range(3,25):

    #Creamos el modelo
    random_forest_model = RandomForestClassifier(n_estimators = 9,  class_weight
                                             max_depth = i, random_state = 4
    random_forest_model.fit(X_train,y_train) #Entrenamos el modelo
    y_train_pred = random_forest_model.predict(X_train) #Prediccion en Train
    y_test_pred = random_forest_model.predict(X_test) #Prediccion en Test
    train_accuracy = accuracy_score(y_train, y_train_pred) #Calculo el accuracy
    test_accuracy = accuracy_score(y_test, y_test_pred) #Calculo el accuracy en

    # Acumulo los resultados de Accuracy para cada corrida:

    A_train.append(train_accuracy)
    A_test.append(test_accuracy)

fig, ax = plt.subplots(figsize=(12, 4))

ax.plot(A_train,marker='o', label='Set Entrenamiento')
ax.plot(A_test,marker='o', label='Set Test', color='C1')

ax.set_xlabel('Máxima profundidad en el árbol para el Random Forest Classifier')
ax.set_xticks(range(0,25,1))
ax.set_ylabel('% Accuracy')
```

```
#ax.set_ylim(0, 1.1)
ax.set_title('Accuracy de un RandomForest')
ax.legend()
```

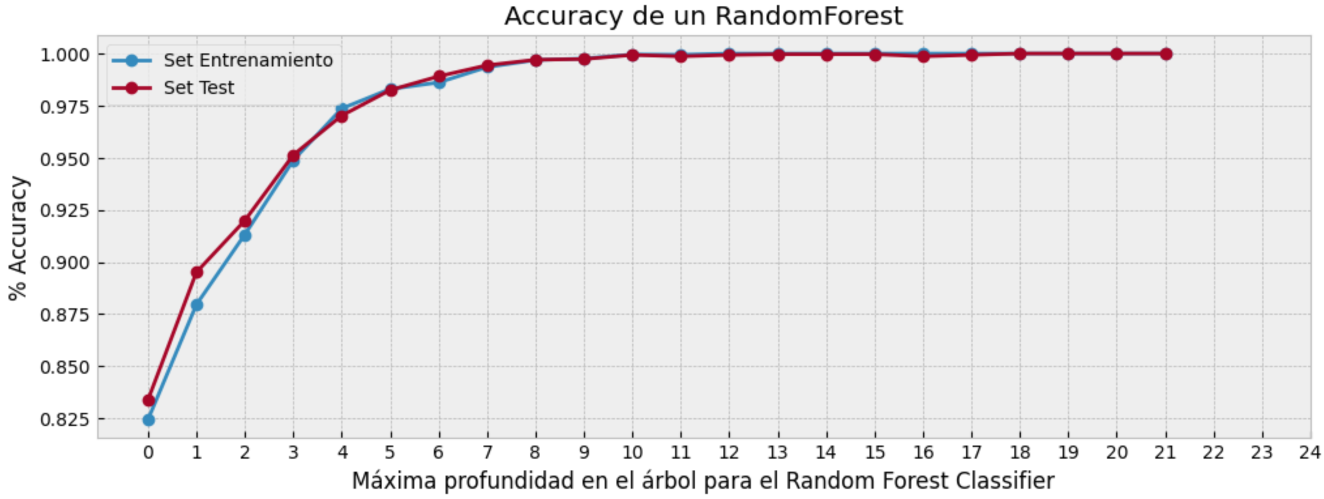
<matplotlib.legend.Legend at 0x7e92c614d090>




```
A_train = []
A_test = []

for i in range(3,25):


    random_forest_model = RandomForestClassifier(n_estimators = 9,  class_weight
                                             max_depth = i, random_state = 4
    random_forest_model.fit(X_train,y_train)
    y_train_pred = random_forest_model.predict(X_train)
    y_test_pred = random_forest_model.predict(X_test)
    train_accuracy = accuracy_score(y_train, y_train_pred)
    test_accuracy = accuracy_score(y_test, y_test_pred) #

    A_train.append(train_accuracy)
    A_test.append(test_accuracy)

fig, ax = plt.subplots(figsize=(12, 4))

ax.plot(A_train,marker='o', label='Set Entrenamiento')
ax.plot(A_test,marker='o', label='Set Test', color='C1')

ax.set_xlabel('Máxima profundidad en el árbol para el Random Forest Classifier')
```

```
ax.set_xticks(range(0,25,1))
ax.set_ylabel('% Accuracy')

ax.set_title('Accuracy de un RandomForest')
ax.legend()
```

```
<matplotlib.legend.Legend at 0x7ac4e16e3460>
```



```
random_forest_model = RandomForestClassifier(n_estimators=9,
                                             max_depth =3,
                                             random_state=42,
                                             max_features="log2")


random_forest_model.fit(X_train, y_train)


y_train_pred_random_forest_model = random_forest_model.predict(X_train)
y_test_pred_random_forest_model = random_forest_model.predict(X_test)


train_accuracy_random_forest = accuracy_score(y_train, y_train_pred_random_fores
test_accuracy_random_forest = accuracy_score(y_test, y_test_pred_random_forest_m

print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_rando
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_random_for
```

```
% de aciertos sobre el set de entrenamiento: 0.918
% de aciertos sobre el set de evaluación: 0.919
```

```
train_accuracy_10_2_2 = accuracy_score(y_train, y_train_pred_random_forest_model

test_accuracy_10_2_2 = accuracy_score(y_test, y_test_pred_random_forest_model)

print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_10_2_
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_10_2_2,3))
```

```
% de aciertos sobre el set de entrenamiento: 0.918
% de aciertos sobre el set de evaluación: 0.919
```

```
confusion_matrix(y_test, y_test_pred_random_forest_model)
```

```
array([[ 393,  237],
       [   9, 2382]])
```

```
precision_10_2_2 = round(precision_score(y_test, y_test_pred_random_forest_model
precision_10_2_2
```

```
0.90951
```

```
recall_10_2_2 = round(recall_score(y_test, y_test_pred_random_forest_model),5)
recall_10_2_2
```

```
0.99624
```

17 Logistic Regression

```
regresion_logistica = LogisticRegression()
```

```
regresion_logistica.fit(X_train, y_train)
```

```
y_train_pred_regresion_logistica = regresion_logistica.predict(X_train)
y_test_pred_regresion_logistica = regresion_logistica.predict(X_test)
```

```
train_accuracy_regresion_logistica = accuracy_score(y_train, y_train_pred_regres
test_accuracy_regresion_logistica = accuracy_score(y_test, y_test_pred_regresion

print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_regre
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_regresion_
```

```
% de aciertos sobre el set de entrenamiento: 0.914
% de aciertos sobre el set de evaluación: 0.912
```

```
train_accuracy_10_3_1 = accuracy_score(y_train, y_train_pred_regresion_logistica
```

```
test_accuracy_10_3_1 = accuracy_score(y_test, y_test_pred_regresion_logistica)

print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_10_3_
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_10_3_1,3))
```

```
% de aciertos sobre el set de entrenamiento: 0.914
% de aciertos sobre el set de evaluación: 0.912
```

```
confusion_matrix(y_test, y_test_pred_regresion_logistica)
```

```
array([[ 432,  198],
       [  69, 2322]])
```

```
precision_10_3_1 = round(precision_score(y_test, y_test_pred_regresion_logistica
precision_10_3_1
```

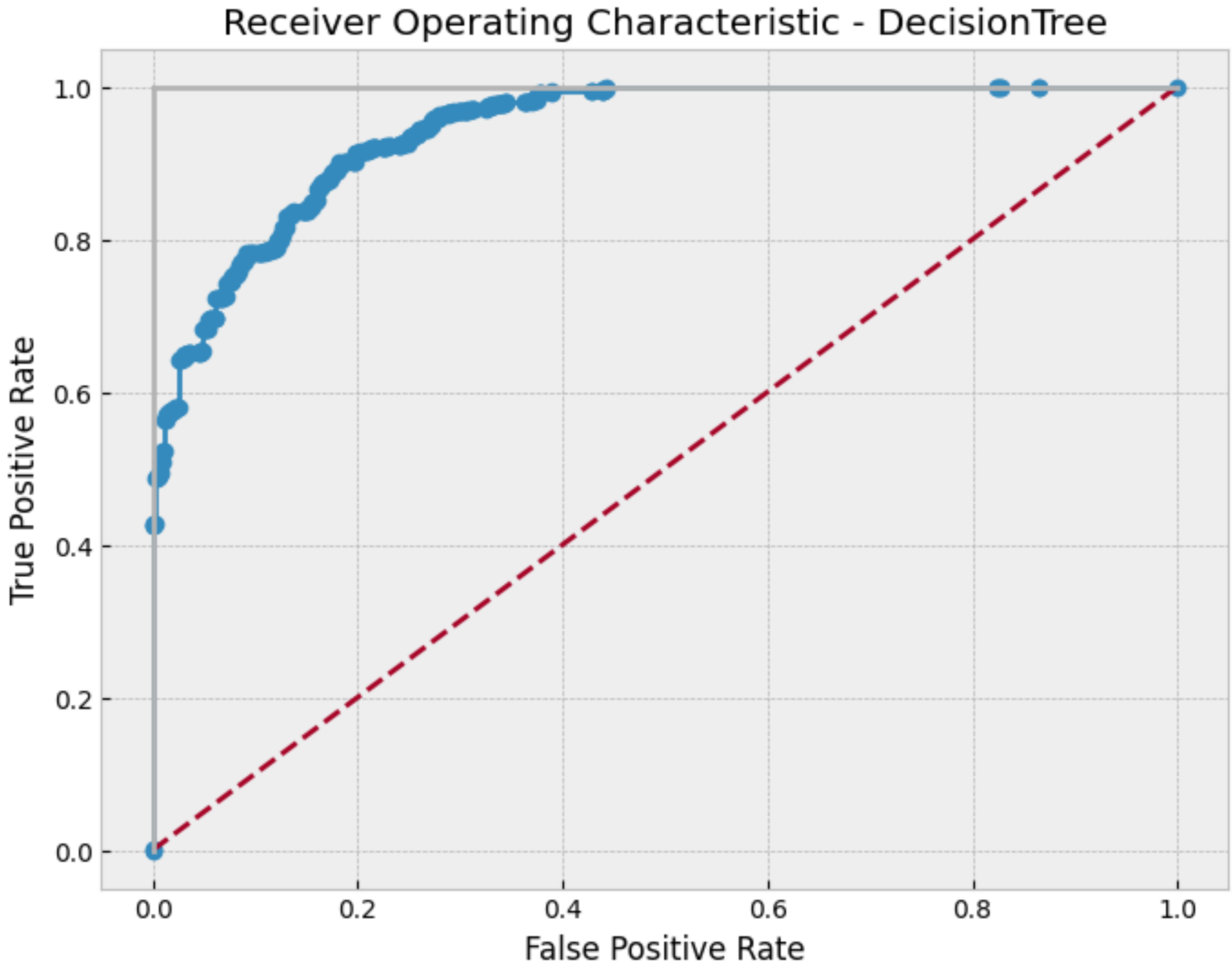
    0.92143

```
recall_10_3_1 = round(recall_score(y_test, y_test_pred_regresion_logistica),5)
recall_10_3_1
```

    0.97114

```
y_score1 = regresion_logistica.predict_proba(X_test)[:,1]

false_positive_rate1, true_positive_rate1, threshold1 = roc_curve(y_test, y_scor
roc_10_3_1 = round(roc_auc_score(y_test, y_score1),5)
print('roc_auc_score for regresion_logistica: ', roc_10_3_1)

plt.subplots(1, figsize=(8,6))
plt.title('Receiver Operating Characteristic - DecisionTree')
plt.plot(false_positive_rate1, true_positive_rate1, marker = 'o')
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

```
roc_auc_score for regresion_logistica:  0.94169
```



```
d3 = {'LogReg':[test_accuracy_10_3_1,precision_10_3_1,recall_10_3_1,roc_10_3_1]}

LogReg = pd.DataFrame(d3, index = ['Accuracy','Precision','Recall','ROC_curve'])
LogReg
```

	LogReg
Accuracy	0.911619
Precision	0.921430
Recall	0.971140
ROC_curve	0.941690

18 KNN

```
classifier = KNeighborsClassifier(n_neighbors = 7, metric = 'minkowski', p = 5)
```

```
classifier.fit(X_train, y_train)
```

```
y_train_pred_knn = classifier.predict(X_train)
y_test_pred_knn = classifier.predict(X_test)
```

```
train_accuracy_knn = accuracy_score(y_train, y_train_pred_knn)
test_accuracy_knn = accuracy_score(y_test, y_test_pred_knn)

print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_knn,3
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_knn,3))
```

```
% de aciertos sobre el set de entrenamiento: 0.994
% de aciertos sobre el set de evaluación: 0.993
```

```
train_accuracy_10_4_1 = accuracy_score(y_train, y_train_pred_knn)

test_accuracy_10_4_1 = accuracy_score(y_test, y_test_pred_knn)

print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_10_4_
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_10_4_1,3))
```

```
% de aciertos sobre el set de entrenamiento: 0.994
% de aciertos sobre el set de evaluación: 0.993
```

```
confusion_matrix(y_test, y_test_pred_knn)
```

```
array([[ 613,   17],
       [   4, 2387]])
```

```
precision_10_4_1 = round(precision_score(y_test, y_test_pred_knn),5)
precision_10_4_1
```

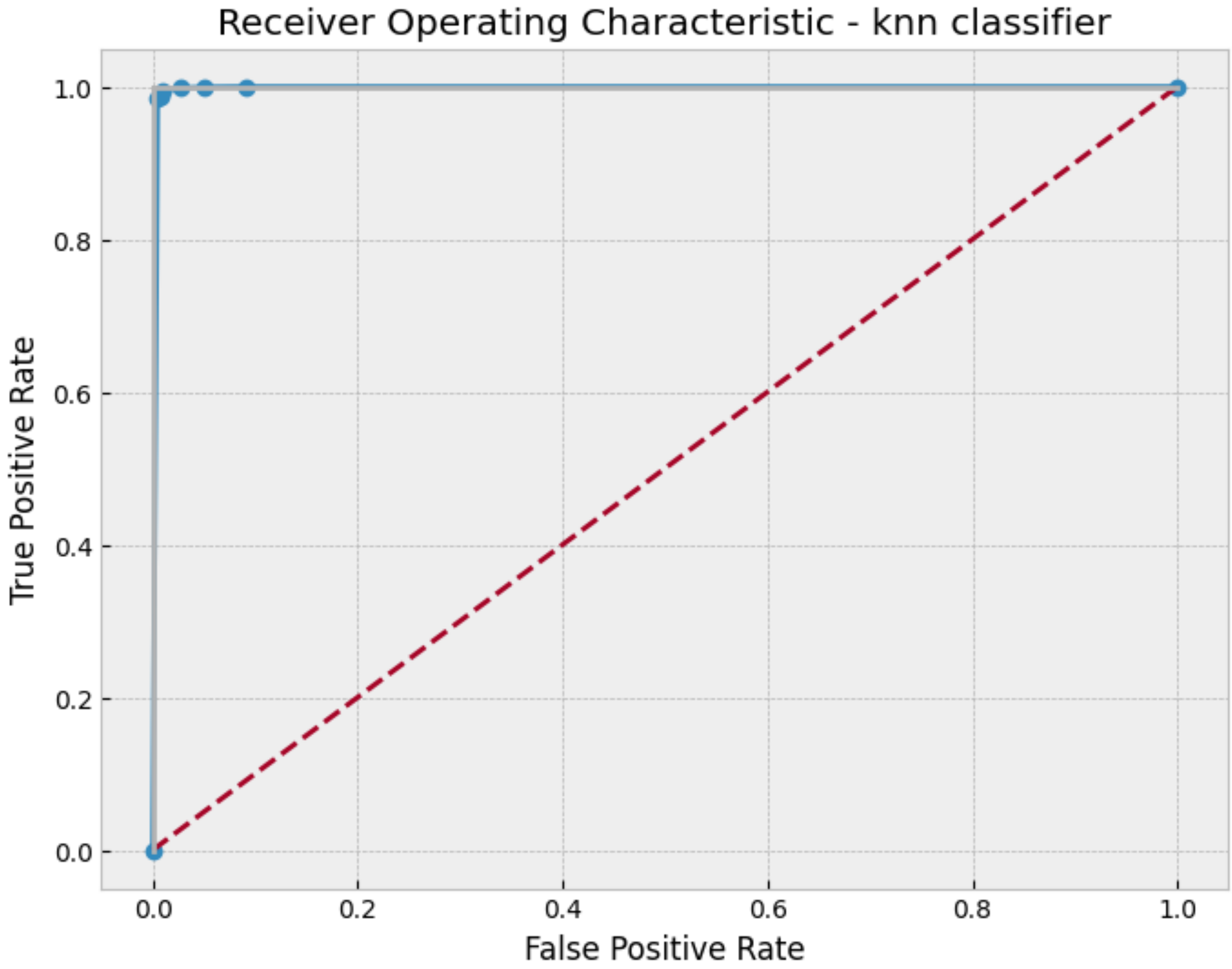
```
0.99293
```

```
recall_10_4_1 = round(recall_score(y_test, y_test_pred_knn),5)
recall_10_4_1
```

```
0.99833
```

```
y_score1 = classifier.predict_proba(X_test)[:,1]

false_positive_rate1, true_positive_rate1, threshold1 = roc_curve(y_test, y_scor
roc_10_4_1 = round(roc_auc_score(y_test, y_score1),5)
print('roc_auc_score for knn classifier: ', roc_10_4_1)

plt.subplots(1, figsize=(8,6))
plt.title('Receiver Operating Characteristic - knn classifier')
plt.plot(false_positive_rate1, true_positive_rate1, marker = 'o')
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

```
roc_auc_score for knn classifier:  0.99742
```



```
d4 = {'knn':[test_accuracy_10_4_1,precision_10_4_1,recall_10_4_1,roc_10_4_1]}

knn_classifier = pd.DataFrame(d4, index = ['Accuracy','Precision','Recall','ROC_
knn_classifier
```

	knn
Accuracy	0.993049
Precision	0.992930
Recall	0.998330
ROC_curve	0.997420

19 Oversampling balanced Árbol de decisión

```
arbol_de_decision = DecisionTreeClassifier(max_depth=4, random_state = 42)
arbol_de_decision.fit(X_train,y_train)
y_train_pred = arbol_de_decision.predict(X_train)
y_test_pred = arbol_de_decision.predict(X_test)
train_accuracy = accuracy_score(y_train, y_train_pred)
test_accuracy = accuracy_score(y_test, y_test_pred)


print('Accuracy:')
train_accuracy_10_6_1_1 = accuracy_score(y_train, y_train_pred)
test_accuracy_10_6_1_1 = accuracy_score(y_test, y_test_pred)
print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_10_6_
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_10_6_1_1,3

print(' Matriz de Confusión')
print(confusion_matrix(y_test, y_test_pred) )
precision_10_6_1_1 = round(precision_score(y_test, y_test_pred),5)
print('\n Precisión:', precision_10_6_1_1)

recall_10_6_1_1 = round(recall_score(y_test, y_test_pred),5)
print('\n Recall: ', recall_10_6_1_1)


y_score1 = arbol_de_decision.predict_proba(X_test)[:,1]

false_positive_rate1, true_positive_rate1, threshold1 = roc_curve(y_test, y_scor
roc_10_6_1_1 = round(roc_auc_score(y_test, y_score1),5)
```

```
print('\n roc_auc_score for DecisionTree: ', roc_10_6_1_1)

plt.subplots(1, figsize=(8,6))
plt.title('Receiver Operating Characteristic - DecisionTree')
plt.plot(false_positive_rate1, true_positive_rate1, marker = 'o')
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

```
Accuracy:
% de aciertos sobre el set de entrenamiento: 0.94
% de aciertos sobre el set de evaluación: 0.942

 Matriz de Confusión
[[ 565   65]
 [ 111 2280]]

 Precisión: 0.97228

 Recall:  0.95358

 roc_auc_score for DecisionTree:  0.97201
```

19.1 Oversampling balanced Logistic Regression

```
regresion_logistica = LogisticRegression()

regresion_logistica.fit(X_train, y_train)

y_train_pred = regresion_logistica.predict(X_train)
y_test_pred = regresion_logistica.predict(X_test)


print('Accuracy:')
train_accuracy_10_6_1_3 = accuracy_score(y_train, y_train_pred)
test_accuracy_10_6_1_3 = accuracy_score(y_test, y_test_pred)
print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_10_6_
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_10_6_1_3,3


print(' Matriz de Confusión')
print(confusion_matrix(y_test, y_test_pred) )


precision_10_6_1_3 = round(precision_score(y_test, y_test_pred),5)
print('\n Precisión:', precision_10_6_1_3)


recall_10_6_1_3 = round(recall_score(y_test, y_test_pred),5)
print('\n Recall: ', recall_10_6_1_3)


y_score1 = arbol_de_decision.predict_proba(X_test)[:,1]

false_positive_rate1, true_positive_rate1, threshold1 = roc_curve(y_test, y_scor
roc_10_6_1_3 = round(roc_auc_score(y_test, y_score1),5)
print('\n roc_auc_score for DecisionTree: ', roc_10_6_1_3)

plt.subplots(1, figsize=(8,6))
plt.title('Receiver Operating Characteristic - Regresion Logistica')
plt.plot(false_positive_rate1, true_positive_rate1, marker = 'o')
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

```
Accuracy:
% de aciertos sobre el set de entrenamiento: 0.914
% de aciertos sobre el set de evaluación: 0.912

 Matriz de Confusión
[[ 432  198]
 [  69 2322]]

 Precisión: 0.92143

 Recall:  0.97114

 roc_auc_score for DecisionTree:  0.97201
```

20 Stratified k-fold Y GridSearch

```
X = df1_1.drop("Do men and women have equal ownership rights to immovable proper

y = df1_1['Do men and women have equal ownership rights to immovable property?']

print('X = ', X.shape)
print('y = ', y.shape)
```

```
X =  (10070, 24)
y =  (10070,)
```

```
skf = StratifiedKFold(n_splits=4, random_state=42, shuffle=True)
```

```
train, test = list(skf.split(X, y))[0]

X_train = X[train]
X_test = X[test]
y_train = y[train]
y_test= y[test]

print('Tamaño de los sets de Test y Train considerando Stratified k-fold Y GridS
print('')
print('X_train = ', X_train.shape)
print('y_train = ', y_train.shape)
print('X_test = ', X_test.shape)
print('y_test = ', y_test.shape)
```

```
Tamaño de los sets de Test y Train considerando Stratified k-fold Y GridSea

X_train =  (7552, 24)
y_train =  (7552,)
X_test =  (2518, 24)
y_test =  (2518,)
```

20.1 Árbol de decisión

```
t = DecisionTreeClassifier(random_state = 42)

par = list(np.arange(0.0, 1., step=0.05))
cv = StratifiedKFold(n_splits=5, random_state=42, shuffle=True)


param_grid = {'max_depth' : list(np.arange(2, 11, step=1)), #np.arange(2,11,1),
             'criterion':['gini', 'entropy'],
             'splitter':['best', 'random'],
             'max_features' : ['auto', 'sqrt', 'log2'],
             'ccp_alpha':par
            }

model = GridSearchCV(estimator=t, param_grid = param_grid,
                     cv=cv, scoring = 'roc_auc') #'accuracy')#,'roc_auc' ])


model.fit(X_train, y_train)
```

```
▸ GridSearchCV
▸ estimator: DecisionTreeClassifier
    ▸ DecisionTreeClassifier
```

```
print("Mejores parametros: "+str(model.best_params_))
print("Mejor Score: "+str(model.best_score_)+'\n')
```

```
Mejores parametros: {'ccp_alpha': 0.0, 'criterion': 'gini', 'max_depth': 1(
Mejor Score: 0.9944949275562018
```

```
arbol_de_decision = DecisionTreeClassifier(random_state = 42,
                                           criterion = 'entropy',
                                           splitter = 'random',
                                           max_depth = 5,
                                           max_features ='auto',
                                           ccp_alpha = 0.0 )


arbol_de_decision.fit(X_train,y_train) #Entrenamos el modelo
y_train_pred = arbol_de_decision.predict(X_train) #Prediccion en Train
y_test_pred = arbol_de_decision.predict(X_test) #Prediccion en Test
```

```
print('Accuracy:')
train_accuracy_10_7_1_1 = accuracy_score(y_train, y_train_pred)
test_accuracy_10_7_1_1 = accuracy_score(y_test, y_test_pred)
print('% de aciertos sobre el set de entrenamiento:', round(train_accuracy_10_7_
print('% de aciertos sobre el set de evaluación:',round(test_accuracy_10_7_1_1,3

print(' Matriz de Confusión')
print(confusion_matrix(y_test, y_test_pred) )


precision_10_7_1_1 = round(precision_score(y_test, y_test_pred),5)
print('\n Precisión:', precision_10_7_1_1)


recall_10_7_1_1 = round(recall_score(y_test, y_test_pred),5)
print('\n Recall: ', recall_10_7_1_1)


y_score1 = arbol_de_decision.predict_proba(X_test)[:,1]

false_positive_rate1, true_positive_rate1, threshold1 = roc_curve(y_test, y_scor
roc_10_7_1_1 = round(roc_auc_score(y_test, y_score1),5)
print('\n roc_auc_score for DecisionTree: ', roc_10_7_1_1)

plt.subplots(1, figsize=(8,6))
plt.title('Receiver Operating Characteristic - DecisionTree')
plt.plot(false_positive_rate1, true_positive_rate1, marker = 'o')
plt.plot([0, 1], ls="--")
plt.plot([0, 0], [1, 0] , c=".7"), plt.plot([1, 1] , c=".7")
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```

```
Accuracy:
% de aciertos sobre el set de entrenamiento: 0.824
% de aciertos sobre el set de evaluación: 0.826

 Matriz de Confusión
[[ 395  133]
 [ 305 1685]]

 Precisión: 0.92684

 Recall:  0.84673

 roc_auc_score for DecisionTree:  0.88786
```

```
d7_1 = {'arbol_optimizado':[test_accuracy_10_7_1_1,precision_10_7_1_1, recall_10

arbol1_sg = pd.DataFrame(d7_1, index = ['Accuracy','Precision','Recall','ROC_cur
arbol1_sg
```

	arbol_optimizado
Accuracy	0.826052
Precision	0.926840
Recall	0.846730
ROC_curve	0.887860

ANALISIS Como en el modelo de regresion linear y por el tipo de variable, si bien es alta la prediccion tambien lo es la tasa de falsos positivos y errores, por lo que debemos centranos en analisis mas evolutivos que de variables en particular.

CONCLUSIONES

Sin dudar han evolucionado los derechos de las mujeres, en mayor medida lo han hecho en países y zonas de mejor situación económica.

Aun falta bastante para que los países lleguen a una igualdad total, es sin duda para replantearse por qué aún nos seguimos preguntando si tenemos iguales Derechos que los hombres.

En ámbitos de países de mejor situación económica, los Derechos han logrado mejorar y esto responde sin dudar a políticas educativas y sociales implantadas a nivel nacional e internacional.

Podemos notar también que al ser variables que son mas de carácter cualitativo, por lo que los análisis numéricos solo van a dar o hacer notar una mínima parte de la realidad. Por lo que hacer conclusiones en base a los datos numéricos para predecir futuras evoluciones, puede llevar a errores.

Al tratarse de datos sociales y de comunidades, las realidades políticas y culturales son un factor que en esta base de datos esta muy poco reflejada, y que a futuro son las que en definitiva tienen el mayor impacto.

Por esto y por todo lo demás es que si bien la tendencia va a ser a la evolución y mejoras de los Derechos de las mujeres, afirmarlo dandole un tiempo preciso es imposible.