




IMPACTO DEL TRABAJO VIA REMOTA

ALEXA ESMERALDA SALDAÑA MORALES

INTRODUCCION A LA CIENCIA DE DATOS

DOCENTE JAIME ALEJANDRO ROMERO SIERRA

Fecha de entrega 27/11/2024



Objetivo de proyecto.

Identificar factores asociados a la salud mental de los trabajadores para buscar mejoras en la atención médica.

¿por qué es importante resolver o estudiar esta problemática?

En el ámbito laboral no es muy tomado en cuenta la salud mental y lo que el ambiente laboral puede llegar a causar en cada uno de los trabajadores. Por ello es importante saber tomar las medidas necesarias y adecuadas para sobrellevar cada uno de los trastornos que hoy en día se han normalizado mucho.

Una buena salud mental hace que el desempeño de cada persona sea diferente de mejor manera, buscamos que los trabajadores puedan desenvolverse adecuadamente en el ámbito laboral, siendo modalidades diferentes hace que la vida cotidiana tenga otro sentido que como comúnmente lo conocemos.

Al analizar los comportamientos y buscar apoyo para cada uno de los individuos, hace que a futuro se tenga un mejor plan de atención con antecedentes que puedan asegurar una mejora eficaz e incluso evitar ese tipo de problemáticas tanto para los trabajadores como para la industria.

Limpieza

Objetivo:

Desarrollar habilidades en el preprocesamiento de datos, incluyendo la identificación y tratamiento de valores faltantes, datos duplicados, y formatos inconsistentes en una base de datos.

Instrucciones:

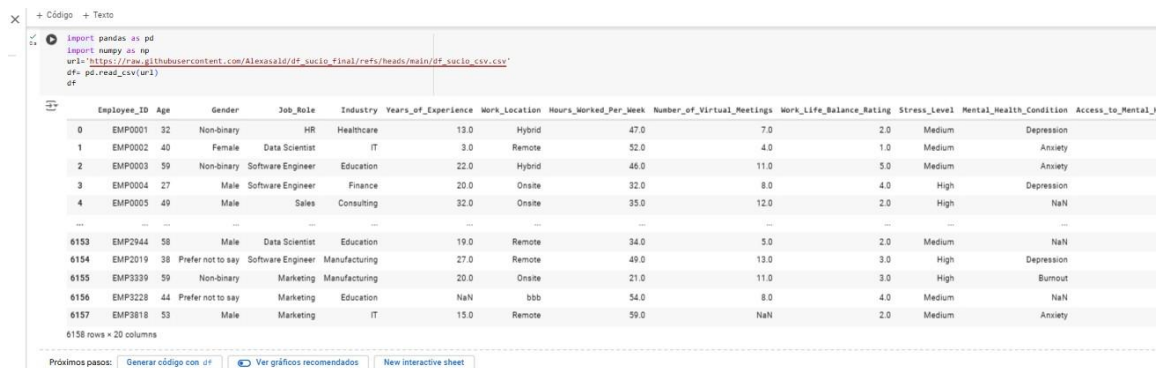
1. Recepción de la Base de Datos

Descargue el archivo de la base de datos "ensuciada" proporcionado por el profesor. Este archivo ha sido alterado con errores comunes, como valores duplicados, valores faltantes (NaNs), y errores en el formato de algunas columnas.

2. Análisis Inicial de la Base de Datos

Antes de comenzar a limpiar la base de datos, debe realizar un análisis preliminar para comprender la naturaleza y distribución de los errores.

- Visualización de la base de datos inicial.
Podemos notar que tenemos un total de 6158 datos, pero existen columnas que mas adelante no serán tan necesarias.



The screenshot shows a Jupyter Notebook interface. The top part contains a code cell with the following Python code:

```
import pandas as pd
import numpy as np
url = "https://raw.githubusercontent.com/Alexsald/df-sucio-final/refs/heads/main/df-sucio_csv.csv"
df = pd.read_csv(url)
df
```

Below the code cell, a preview of the dataset is displayed as a table. The table has 13 columns: Employee_ID, Age, Gender, Job_Role, Industry, Years_of_Experience, Work_Location, Hours_Worked_Per_Week, Number_of_Virtual_Meetings, Work-Life_Balance_Rating, Stress_Level, Mental_Health_Condition, and Access_to_Mental_He. The first few rows are shown, followed by an ellipsis, and then rows 6153 through 6157. The bottom of the preview indicates "6158 rows x 13 columns". At the very bottom, there are three buttons: "Generar código con df", "Ver gráficos recomendados", and "New interactive sheet".

	Employee_ID	Age	Gender	Job_Role	Industry	Years_of_Experience	Work_Location	Hours_Worked_Per_Week	Number_of_Virtual_Meetings	Work-Life_Balance_Rating	Stress_Level	Mental_Health_Condition	Access_to_Mental_He
0	EMP0001	32	Non-binary	HR	Healthcare	13.0	Hybrid	47.0	7.0	2.0	Medium	Depression	
1	EMP0002	40	Female	Data Scientist	IT	3.0	Remote	52.0	4.0	1.0	Medium	Anxiety	
2	EMP0003	59	Non-binary	Software Engineer	Education	22.0	Hybrid	46.0	11.0	5.0	Medium	Anxiety	
3	EMP0004	27	Male	Software Engineer	Finance	20.0	Onsite	32.0	8.0	4.0	High	Depression	
4	EMP0005	49	Male	Sales	Consulting	32.0	Onsite	35.0	12.0	2.0	High	NaN	
...
6153	EMP2944	58	Male	Data Scientist	Education	19.0	Remote	34.0	5.0	2.0	Medium	NaN	
6154	EMP2019	38	Prefer not to say	Software Engineer	Manufacturing	27.0	Remote	49.0	13.0	3.0	High	Depression	
6155	EMP3339	59	Non-binary	Marketing	Manufacturing	20.0	Onsite	21.0	11.0	3.0	High	Burnout	
6156	EMP3228	44	Prefer not to say	Marketing	Education	NaN	bbb	54.0	8.0	4.0	Medium	NaN	
6157	EMP3818	53	Male	Marketing	IT	15.0	Remote	59.0	NaN	2.0	Medium	Anxiety	

- Resumen estadístico de los datos.

con el comando de describe(), podemos visualizar la tabla con el conteo completo en casi todas las columnas excepto en la de actividad física. También se muestran la falta de columnas ya que el tipo de dato no concuerda con ser dato numérico por ello más adelante tendremos que modificar estos datos.

```
[450] #resumen estadístico de los datos
df.describe()
```

	Años_Expe	Hrs_semana	N_reun_V	Balance_vida_trabajo	Calificación_aislamientoS	Actividad_fisica
count	5851.000000	5851.000000	5851.000000	5851.000000	5851.000000	0.0
mean	17.900530	39.663818	7.558879	2.988378	3.002905	NaN
std	10.050676	11.872569	4.643983	1.413803	1.398103	NaN
min	1.000000	20.000000	0.000000	1.000000	1.000000	NaN
25%	9.000000	29.000000	4.000000	2.000000	2.000000	NaN
50%	18.000000	40.000000	8.000000	3.000000	3.000000	NaN
75%	26.000000	50.000000	12.000000	4.000000	4.000000	NaN
max	35.000000	60.000000	15.000000	5.000000	5.000000	NaN

- Porcentaje de valores faltantes por columna

El siguiente comando. isnull().mean(), muestra la tabla de porcentajes de cada columna, aquí podemos interpretar que en su mayoría los porcentajes de datos faltantes son del 4.9854% , la de salud mental es de 27.72% y la de actividad física de un 100% lo cual es preocupante ya que sus datos son nulos.

```
#porcentaje de valores faltantes inicial
df.isnull().mean()
```

ID_E	0.049854
Edad	0.049854
Genero	0.049854
Rol	0.049854
Industria	0.049854
Años_Expe	0.049854
Ubicacion	0.049854
Hrs_semana	0.049854
N_reun_V	0.049854
Balance_vida_trabajo	0.049854
Nivel_estres	0.049854
Salud_Mental	0.277200
Recurso_SaludM	0.049854
Cambio_en_productividad	0.049854
Calificación_aislamientoS	0.049854
Satisfacción_trabajo	0.049854
Apoyo_de_empresa	0.049854
Actividad_fisica	1.000000
Calidad_del_sueño	0.049854
Region	0.049854

dtype: float64

• Existencia duplicados

Con el comando. `df.duplicated().sum` Se muestra el conteo total de filas duplicadas

```
✓ [452] #Contar filas duplicadas
df.duplicated().sum()
145
```

• Tipo de dato

En la siguiente tabla observamos lo mencionado anteriormente, los datos están definidos de manera incorrecta lo cual nos provocara algunos conflictos para el análisis.

```
✓ [333] #tipo de datos
df1.info()

<class 'pandas.core.frame.DataFrame'>
Index: 6013 entries, 0 to 6157
Data columns (total 20 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   ID_E                  5706 non-null  object
1   Edad                 5707 non-null  object
2   Genero               5708 non-null  object
3   Rol                  5706 non-null  object
4   Industria            5706 non-null  object
5   Años_Expe            5706 non-null  float64
6   Ubicacion            5707 non-null  object
7   Hrs_semana           5706 non-null  float64
8   N_reun_V             5707 non-null  float64
9   Balance_vida_trabajo 5707 non-null  float64
10  Nivel_estres          5706 non-null  object
11  Salud_Mental          4340 non-null  object
12  Recurso_SaludM        5708 non-null  object
13  Cambio_en_productividad 5706 non-null  object
14  Calificación_aislamientos 5706 non-null  float64
15  Satisfacción_trabajo  5707 non-null  object
16  Apoyo_de_empresa      5707 non-null  object
17  Actividad_fisica      0 non-null    float64
18  Calidad_del_sueño     5707 non-null  object
19  Region                5706 non-null  object
dtypes: float64(6), object(14)
memory usage: 986.5+ KB
```

Ahora al conocer un poco mas los datos con los que se trabajara y los detalles que se tendrán que tomar en cuenta, iniciare con el proceso de limpieza teniendo en cuenta el nuevo etiquetado de datos y la resolución de los datos faltantes.

Uno de los problemas es que la cantidad de datos faltantes es importante ya que en la siguiente fase de limpieza suelen perderse muchos de los datos, haciendo así la base de datos muy pequeña.

Limpieza de datos.

- **Eliminación de duplicados**

Agregue un nuevo DF1 para no trabajar sobre el mismo, quite los duplicados de todo el DF, anteriormente teníamos un total de 6158 y ahora eliminando duplicados

```
df1=df.drop_duplicates()
df1
```

	ID_E	Edad	Genero	Rol	Industria	Años_Expe	Ubicacion	Hrs_sesana	N_reun_V	Balance_vida_trabajo	Nivel_estres	Salud_Mental	Recurso_SaludM	Cambio_en_productividad	Calif
0	EMP0001	32	Non-binary	HR	Healthcare	13.0	Hybrid	47.0	7.0	2.0	Medium	Depression	NaN	bbb	
1	EMP0002	40	Female	Data Scientist	IT	3.0	Remote	32.0	4.0	1.0	Medium	Anxiety	NaN	Increase	
2	EMP0003	59	Non-binary	Software Engineer	Education	22.0	Hybrid	46.0	11.0	5.0	Medium	Anxiety	No	No Change	
3	EMP0004	27	Male	Software Engineer	Finance	20.0	Onsite	32.0	8.0	4.0	High	Depression	Yes	Increase	
4	EMP0005	49	Male	Sales	Consulting	32.0	Onsite	35.0	12.0	2.0	High	NaN	Yes	NaN	
...
6152	EMP1138	28	Female	Marketing	NaN	5.0	Onsite	52.0	0.0	4.0	Low	Burnout	Yes	Increase	
6153	EMP2944	58	Male	Data Scientist	Education	19.0	Remote	34.0	5.0	2.0	Medium	NaN	No	NaN	
6155	EMP3339	59	Non-binary	Marketing	Manufacturing	20.0	Onsite	21.0	11.0	3.0	High	Burnout	No	Increase	
6156	EMP3228	44	Prefer not to say	Marketing	Education	NaN	bbb	54.0	8.0	4.0	Medium	NaN	Yes	Decrease	
6157	EMP3516	33	Male	Marketing	IT	15.0	Remote	59.0	NaN	2.0	Medium	Anxiety	bbb	No Change	

6013 rows × 20 columns

- **Corrección de datos 'bbb'**

Coloque un ciclo para saber cuántos datos había escritos de esa manera, después realice otro ciclo for para eliminar aquellos datos inválidos. Dejando así solo datos NaN y los datos normales para después rellenar los datos vacíos. Nota (profesor intentó convertirlos a datos NaN pero me las cadenas de caracteres no me dejaba colocarlos a datos nulos)

```
[336] #numero de veces que esta 'bbb'
for i in lista:
    print(f'En la columna {i} la cadena de 'bbb' son: {df2[df2[i] == 'bbb'].shape[0]}')

En la columna ID_E la cadena de 'bbb' son: 0
En la columna Edad la cadena de 'bbb' son: 118
En la columna Genero la cadena de 'bbb' son: 0
En la columna Rol la cadena de 'bbb' son: 115
En la columna Industria la cadena de 'bbb' son: 0
En la columna Años_Expe la cadena de 'bbb' son: 0
En la columna Ubicacion la cadena de 'bbb' son: 115
En la columna Hrs_semana la cadena de 'bbb' son: 0
En la columna N_reun_V la cadena de 'bbb' son: 0
En la columna Balance_vida_trabajo la cadena de 'bbb' son: 0
En la columna Nivel_estres la cadena de 'bbb' son: 117
En la columna Salud_Mental la cadena de 'bbb' son: 115
En la columna Recurso_SaludM la cadena de 'bbb' son: 118
En la columna Cambio_en_productividad la cadena de 'bbb' son: 119
En la columna Calificación_alimentación la cadena de 'bbb' son: 0
En la columna Satisfacción_trabajo la cadena de 'bbb' son: 0
En la columna Apoyo_de_empresa la cadena de 'bbb' son: 111
En la columna Actividad_fisica la cadena de 'bbb' son: 0
En la columna Calidad_del_sueño la cadena de 'bbb' son: 114
En la columna Region la cadena de 'bbb' son: 0

#Ciclo para eliminar 'bbb'
for i in lista:
    df2=df2[df2[i] != 'bbb']
df2
```

	ID_E	Edad	Genero	Rol	Industria	Años_Expe	Ubicacion	Hrs_semana	N_reun_V	Balance_vida_trabajo	Nivel_estres	Salud_Mental	Recurso_SaludM	Can
1	EMP0002	40	Female	Data Scientist	IT	3.0	Remote	52.0	4.0	1.0	Medium	Anxiety	NaN	
2	EMP0003	59	Non-binary	Software Engineer	Education	22.0	Hybrid	46.0	11.0	5.0	Medium	Anxiety	No	
3	EMP0004	27	Male	Software Engineer	Finance	20.0	Onsite	32.0	8.0	4.0	High	Depression	Yes	
4	EMP0005	49	Male	Sales	Consulting	32.0	Onsite	35.0	12.0	2.0	High	NaN	Yes	
5	EMP0006	59	Non-binary	Sales	IT	31.0	Hybrid	39.0	3.0	4.0	High	NaN	No	
...
6149	EMP4544	56	Prefer not to say	Software Engineer	Education	5.0	Hybrid	35.0	3.0	5.0	Medium	Anxiety	Yes	
6151	EMP3533	41	Prefer not to say	Project Manager	IT	9.0	Onsite	50.0	8.0	2.0	High	NaN	Yes	
6152	EMP1138	28	Female	Marketing	NaN	5.0	Onsite	52.0	0.0	4.0	Low	Burnout	Yes	
6153	EMP2944	58	Male	Data Scientist	Education	19.0	Remote	34.0	5.0	2.0	Medium	NaN	No	
6155	EMP3339	59	Non-binary	Marketing	Manufacturing	20.0	Onsite	21.0	11.0	3.0	High	Burnout	No	

5040 rows x 15 columns

• Corrección de tipos de datos:

Convertir solamente a datos numéricos aquellos que estaban etiquetados como object para poder usarlos como números y se reconocieran de esa manera.

```
[460] #cambio de tipo de dato a numerico
df3['Edad']=df3['Edad'].astype(float)
df3['Años_Expe']=df3['Años_Expe'].astype(float)
df3['Hrs_semana']=df3['Hrs_semana'].astype(float)
df3['Apoyo_de_empresa']=df3['Apoyo_de_empresa'].astype(float)
df3['Balance_vida_trabajo']=df3['Balance_vida_trabajo'].astype(float)
```

• Eliminación de fila con datos Nan

```
<ipython-input-460-25b2526f83fb>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df3['Edad']=df3['Edad'].astype(float)
<ipython-input-460-25b2526f83fb>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df3['Apoyo_de_empresa']=df3['Apoyo_de_empresa'].astype(float)
```

```
df3['Edad']=df3['Edad'].astype(int)
df3['Años_Expe']=df3['Años_Expe'].astype(int)
df3['Hrs_semana']=df3['Hrs_semana'].astype(int)
df3['Apoyo_de_empresa']=df3['Apoyo_de_empresa'].astype(int)
df3['Balance_vida_trabajo']=df3['Balance_vida_trabajo'].astype(int)
```

Como lo vimos anteriormente, la fila de actividad física tenía el 100% de datos faltantes por lo que decidí eliminarla, junto con el ID del empleado lo cual no se hace relevante en el análisis futuro

```
#elimino la columna de 'actividad_fisica' porque esta completaamete vacia.
df3= df2.drop(columns=['ID_E', 'Actividad_fisica', 'H_reun_V',])
df3
```

✓ 0.0s

	Edad	Genero	Rol	Industria	Años_Expe	Ubicacion	Hrs_semana	Balance_vida_trabajo	Nivel_estres	Salud_Mental	Recurso_SaludM	Cambio_en_productividad	Califi
1	40.0	Female	Data Scientist	IT	3.0	Remote	52.0	1.0	Medium	Anxiety	NaN	Increase	
2	59.0	Non-binary	Software Engineer	Education	22.0	Hybrid	46.0	5.0	Medium	Anxiety	No	No Change	
3	27.0	Male	Software Engineer	Finance	20.0	Onsite	32.0	4.0	High	Depression	Yes	Increase	
4	49.0	Male	Sales	Consulting	32.0	Onsite	35.0	2.0	High	NaN	Yes	NaN	
5	59.0	Non-binary	Sales	IT	31.0	Hybrid	39.0	4.0	High	NaN	No	Increase	
...
6149	56.0	Prefer not to say	Software Engineer	Education	5.0	Hybrid	35.0	5.0	Medium	Anxiety	Yes	Increase	
6151	41.0	Prefer not to say	Project Manager	IT	9.0	Onsite	50.0	2.0	High	NaN	Yes	No Change	
6152	28.0	Female	Marketing	NaN	5.0	Onsite	52.0	4.0	Low	Burnout	Yes	Increase	
6153	58.0	Male	Data Scientist	Education	19.0	Remote	34.0	2.0	Medium	NaN	No	NaN	
6155	59.0	Non-binary	Marketing	Manufacturing	20.0	Onsite	21.0	3.0	High	Burnout	No	Increase	

5040 rows x 17 columns

- **Eliminación o imputación de valores faltantes:**

Rellene los valores nulos por moda y promedio, así no serían muchos datos eliminados o perdidos. Ya que como se vio en la tabla de porcentajes, era una cantidad considerable de datos faltantes lo que haría que el análisis fuera complicado y no muy preciso.

Al realizar

```
#Rellenar datos faltan
```

```
df3['Edad']=df3['Edad'].fillna(df3['Edad'].mean())
df3['Genero']=df3['Genero'].fillna(df3['Genero'].mode()[0])
df3['Nivel_estres']=df3['Nivel_estres'].fillna(df3['Nivel_estres'].mode()[0])
df3['Rol']=df3['Rol'].fillna(df3['Rol'].mode()[0])
df3['Años_Expe']=df3['Años_Expe'].fillna(df3['Años_Expe'].mean())
df3['Hrs_semana']=df3['Hrs_semana'].fillna(df3['Hrs_semana'].mean())
df3['Balance_vida_trabajo']=df3['Balance_vida_trabajo'].fillna(df3['Balance_vida_trabajo'].mean())
df3['Salud_Mental']=df3['Salud_Mental'].fillna(df3['Salud_Mental'].mode()[0])
df3['Industria']=df3['Industria'].fillna(df3['Industria'].mode()[0])
df3['Recurso_SaludM']=df3['Recurso_SaludM'].fillna(df3['Recurso_SaludM'].mode()[0])
df3['Ubicacion']=df3['Ubicacion'].fillna(df3['Ubicacion'].mode()[0])
df3['Region']=df3['Region'].fillna(df3['Region'].mode()[0])
df3['Cambio_en_productividad']=df3['Cambio_en_productividad'].fillna(df3['Cambio_en_productividad'].mode()[0])
df3['Calificación_aislamientoS']=df3['Calificación_aislamientoS'].fillna(df3['Calificación_aislamientoS'].mode()[0])
df3['Satisfacción_trabajo']=df3['Satisfacción_trabajo'].fillna(df3['Satisfacción_trabajo'].mode()[0])
df3['Apoyo_de_empresa']=df3['Apoyo_de_empresa'].fillna(df3['Apoyo_de_empresa'].mode()[0])
df3['Calidad_del_sueño']=df3['Calidad_del_sueño'].fillna(df3['Calidad_del_sueño'].mode()[0])
```

df3

✓ 0.0s

	Edad	Genero	Rol	Industria	Años_Expe	Ubicacion	Hrs_semana	Balance_vida_trabajo	Nivel_estres	Salud_Mental	Recurso_SaludM	Cambio_en_productividad	Calificación_aislamientoS	Satisfacción_trabajo	Apoyo_de_empresa	Calidad_del_sueño
1	40.0	Female	Data Scientist	IT	3.0	Remote	52.0	1.0	Medium	Anxiety	No	Increase	3.0	Satisfied		
2	59.0	Non-binary	Software Engineer	Education	22.0	Hybrid	46.0	5.0	Medium	Anxiety	No	No Change	4.0	Unsatisfied		
3	27.0	Male	Software Engineer	Finance	20.0	Onsite	32.0	4.0	High	Depression	Yes	Increase	3.0	Unsatisfied		
4	49.0	Male	Sales	Consulting	32.0	Onsite	35.0	2.0	High	Anxiety	Yes	Decrease	3.0	Unsatisfied		
5	59.0	Non-binary	Sales	IT	31.0	Hybrid	39.0	4.0	High	Anxiety	No	Increase	5.0	Unsatisfied		
...
6149	56.0	Prefer not to say	Software Engineer	Education	5.0	Hybrid	35.0	5.0	Medium	Anxiety	Yes	Increase	2.0	Neutral		
6151	41.0	Prefer not to say	Project Manager	IT	9.0	Onsite	50.0	2.0	High	Anxiety	Yes	No Change	5.0	Neutral		
6152	28.0	Female	Marketing	Retail	5.0	Onsite	52.0	4.0	Low	Burnout	Yes	Increase	1.0	Unsatisfied		
6153	58.0	Male	Data Scientist	Education	19.0	Remote	34.0	2.0	Medium	Anxiety	No	Decrease	5.0	Satisfied		
6155	59.0	Non-binary	Marketing	Manufacturing	20.0	Onsite	21.0	3.0	High	Burnout	No	Increase	1.0	Neutral		

5040 rows x 17 columns

Resultados.

En conclusión, la base de datos ha sido trabajada en un proceso de limpieza, pudimos ver los conflictos que había en ella. Los datos ahora nos permitirán hacer el análisis correcto, sin embargo el análisis pudiera no llegar a ser 100% confiable ya que en el transcurso de limpieza siempre llegan a perderse algunos datos.

Tipo de dato final

```
df3.info()
[121] ✓ 0.0s
...
<class 'pandas.core.frame.DataFrame'>
Index: 5040 entries, 1 to 6155
Data columns (total 17 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Edad                                 5040 non-null   int64
1   Genero                             5040 non-null   object
2   Rol                                 5040 non-null   object
3   Industria                          5040 non-null   object
4   Años_Expe                          5040 non-null   int64
5   Ubicacion                          5040 non-null   object
6   Hrs_semana                         5040 non-null   int64
7   Balance_vida_trabajo              5040 non-null   int64
8   Nivel_estres                      5040 non-null   object
9   Salud_Mental                     5040 non-null   object
10  Recurso_SaludM                   5040 non-null   object
11  Cambio_en_productividad          5040 non-null   object
12  Calificación_aislamientoS        5040 non-null   float64
13  Satisfacción_trabajo             5040 non-null   object
14  Apoyo_de_empresa                 5040 non-null   int64
15  Calidad_del_sueño                5040 non-null   object
16  Region                           5040 non-null   object
dtypes: float64(1), int64(5), object(11)
memory usage: 708.8+ KB
```

Tabla de datos faltantes, a comparación de la tabla que se presentó al inicio con porcentajes un tanto altos en cada columna, en la actual podemos observar que no existen datos faltantes.

```
df3.isnull().mean()
[97] ✓ 0.0s
...
Edad                                0.0
Genero                             0.0
Rol                                 0.0
Industria                          0.0
Años_Expe                          0.0
Ubicacion                          0.0
Hrs_semana                         0.0
Balance_vida_trabajo              0.0
Nivel_estres                      0.0
Salud_Mental                     0.0
Recurso_SaludM                   0.0
Cambio_en_productividad          0.0
Calificación_aislamientoS        0.0
Satisfacción_trabajo             0.0
Apoyo_de_empresa                 0.0
Calidad_del_sueño                0.0
Region                           0.0
dtype: float64
```

Datos inválidos

De igual manera los datos 'bbb' ya se encuentran eliminados de la base de datos

```
#cantidad de 'bbb' encontrados despues de a eliminación
for i in lista:
    print(f"En la columna {i} la cadena de 'bbb' son: {df3[df3[i] == 'bbb'].shape[0]}")

22] ✓ 0.0s

En la columna ID_E la cadena de 'bbb' son: 0
En la columna Edad la cadena de 'bbb' son: 0
En la columna Genero la cadena de 'bbb' son: 0
En la columna Rol la cadena de 'bbb' son: 0
En la columna Industria la cadena de 'bbb' son: 0
En la columna Años_Expe la cadena de 'bbb' son: 0
En la columna Ubicacion la cadena de 'bbb' son: 0
En la columna Hrs_semana la cadena de 'bbb' son: 0
En la columna M_reun_V la cadena de 'bbb' son: 0
En la columna Balance_vida_trabajo la cadena de 'bbb' son: 0
En la columna Nivel_estres la cadena de 'bbb' son: 0
En la columna Salud_Mental la cadena de 'bbb' son: 0
En la columna Recurso_SaludM la cadena de 'bbb' son: 0
En la columna Cambio_en_productividad la cadena de 'bbb' son: 0
En la columna Calificación_aislamientoS la cadena de 'bbb' son: 0
En la columna Satisfacción_trabajo la cadena de 'bbb' son: 0
En la columna Apoyo_de_empresa la cadena de 'bbb' son: 0
En la columna Actividad_fisica la cadena de 'bbb' son: 0
En la columna Calidad_del_sueño la cadena de 'bbb' son: 0
En la columna Region la cadena de 'bbb' son: 0
```

EDA

Visión general

Dataset

VISION GENERAL

```
[246] #Número total de registros y variables
      f"{df.shape[0]} filas y {df.shape[1]} columnas"
```

```
🔍 '5040 filas y 18 columnas'
```

```
[247] df.columns
```

```
🔍 Index(['Edad', 'Genero', 'Rol', 'Industria', 'Años_Expe', 'Ubicacion',
        'Hrs_semana', 'Balance_vida_trabajo', 'Nivel_estres', 'Salud_Mental',
        'Recurso_SaludM', 'Cambio_en_productividad',
        'Calificación_aislamientos', 'Satisfacción_trabajo', 'Apoyo_de_empresa',
        'Calidad_del_sueño', 'Region', 'Genero_num'],
        dtype='object')
```

```
#clasificacion de variables
categoricas=df.select_dtypes(exclude='number')
numericas=df.select_dtypes(include='number')
print("variable categoricas")
print("\nCategorica:Datos que describen un grupo o categorias sin orden")
print(categoricas)
print("variable numericas")
print("\nNumericas:Datos que representan un valor medible o calculable ")
print(numericas)
```

```
🔍 variable categoricas
```

Categorica:Datos que describen un grupo o categorias sin orden

	Genero	Rol	Industria	Ubicacion
0	Female	Data Scientist	IT	Remote
1	Non-binary	Software Engineer	Education	Hybrid
2	Male	Software Engineer	Finance	Onsite
3	Male	Sales	Consulting	Onsite
4	Non-binary	Sales	IT	Hybrid
...
5035	Prefer not to say	Software Engineer	Education	Hybrid
5036	Prefer not to say	Project Manager	IT	Onsite
5037	Female	Marketing	Retail	Onsite
5038	Male	Data Scientist	Education	Remote
5039	Non-binary	Marketing	Manufacturing	Onsite

	Nivel_estres	Salud_Mental	Recurso_SaludM	Cambio_en_productividad
0	Medium	Anxiety	No	Increase
1	Medium	Anxiety	No	No Change
2	High	Depression	Yes	Increase
3	High	Anxiety	Yes	Decrease
4	High	Anxiety	No	Increase
...
5035	Medium	Anxiety	Yes	Increase
5036	High	Anxiety	Yes	No Change
5037	Low	Burnout	Yes	Increase
5038	Medium	Anxiety	No	Decrease
5039	High	Burnout	No	Increase

	Satisfacción_trabajo	Calidad_del_sueño	Region
0	Satisfied	Good	Asia
1	Unsatisfied	Poor	North America
2	Unsatisfied	Poor	Europe
3	Unsatisfied	Average	North America
4	Unsatisfied	Average	South America
...
5035	Neutral	Average	Asia
5036	Neutral	Average	Africa
5037	Unsatisfied	Poor	Asia
5038	Satisfied	Good	North America
5039	Neutral	Average	Africa

[5040 rows x 11 columns]

variable numericas

Numericas:Datos que representan un valor medible o calculable

	Edad	Años_Expe	Hrs_semana	Balance_vida_trabajo	\
0	40.0	3.0	52.0	1.0	
1	59.0	22.0	46.0	5.0	
2	27.0	20.0	32.0	4.0	
3	49.0	32.0	35.0	2.0	
4	59.0	31.0	39.0	4.0	
...	
5035	56.0	5.0	35.0	5.0	
5036	41.0	9.0	50.0	2.0	
5037	28.0	5.0	52.0	4.0	
5038	58.0	19.0	34.0	2.0	
5039	59.0	20.0	21.0	3.0	

	Calificación_aislamientos	Apoyo_de_empresa	Genero_num
0	3.0	2.0	1
1	4.0	5.0	3
2	3.0	3.0	2
3	3.0	3.0	2
4	5.0	1.0	3
...
5035	2.0	5.0	4
5036	5.0	5.0	4
5037	1.0	1.0	1
5038	5.0	3.0	2
5039	1.0	1.0	3

[5040 rows x 7 columns]

Realice cambios en el tipo de datos para que mas adelante se pudiera realizar la estadística correctamente

```
[249] df=df
df['Edad']=df['Edad'].astype(float)
df['Años_Expe']=df['Años_Expe'].astype(float)
df['Hrs_semana']=df['Hrs_semana'].astype(float)
df['Balance_vida_trabajo']=df['Balance_vida_trabajo'].astype(float)
df['Apoyo_de_empresa']=df['Apoyo_de_empresa'].astype(float)
```

#Tipo de dato
df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5040 entries, 0 to 5039
Data columns (total 18 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Edad                  5040 non-null  float64
1   Genero                 5040 non-null  object
2   Rol                    5040 non-null  object
3   Industria              5040 non-null  object
4   Años_Expe              5040 non-null  float64
5   Ubicacion              5040 non-null  object
6   Hrs_semana             5040 non-null  float64
7   Balance_vida_trabajo   5040 non-null  float64
8   Nivel_estres           5040 non-null  object
9   Salud_Mental           5040 non-null  object
10  Recurso_SaludM         5040 non-null  object
11  Cambio_en_productividad 5040 non-null  object
12  Calificación_aislamientos 5040 non-null  float64
13  Satisfacción_trabajo    5040 non-null  object
14  Apoyo_de_empresa       5040 non-null  float64
15  Calidad_del_sueño       5040 non-null  object
16  Region                 5040 non-null  object
17  Genero_num             5040 non-null  int64
dtypes: float64(6), int64(1), object(11)
memory usage: 788.9+ KB
```

[251] df.describe()

	Edad	Años_Expe	Hrs_semana	Balance_vida_trabajo	Calificación_aislamientos	Apoyo_de_empresa	Genero_num
count	5040.000000	5040.000000	5040.000000	5040.000000	5040.000000	5040.000000	5040.000000
mean	41.634325	17.724008	39.537103	2.932341	2.959722	2.992857	2.463294
std	11.353221	9.762562	11.510619	1.394567	1.382195	1.367103	1.091671
min	22.000000	1.000000	20.000000	1.000000	1.000000	1.000000	1.000000
25%	32.000000	9.000000	30.000000	2.000000	2.000000	2.000000	2.000000
50%	42.000000	17.000000	39.000000	3.000000	3.000000	3.000000	2.000000
75%	52.000000	26.000000	49.000000	4.000000	4.000000	4.000000	3.000000
max	60.000000	35.000000	60.000000	5.000000	5.000000	5.000000	4.000000

Resumen estadístico

RESUMEN ESTADISTICO

[252] #Resumen estadisticasa descriptivas
df1=df
#variables numericas
n=df.select_dtypes(include='number')
numericas=n.describe().T
numericas['varianza']=n.var()
#variables categoricas
categoricas=df.select_dtypes(exclude='number')
resumencate={col: df[col].value_counts().sort_values(ascending=False) for col in categoricas}

print("Resumen de variables numericas")
print(numericas)
print("\nResumen de variables categoricas")
for col,freq in resumencate.items():
print(f"{col}:\n{freq}\n")

Resumen de variables numericas

	count	mean	std	min	25%	50%	\
Edad	5040.0	41.634325	11.353221	22.0	32.0	42.0	
Años_Expe	5040.0	17.724008	9.762562	1.0	9.0	17.0	
Hrs_semana	5040.0	39.537103	11.510619	20.0	30.0	39.0	
Balance_vida_trabajo	5040.0	2.932341	1.394567	1.0	2.0	3.0	
Calificación_aislamientos	5040.0	2.959722	1.382195	1.0	2.0	3.0	
Apoyo_de_empresa	5040.0	2.992857	1.367103	1.0	2.0	3.0	
Genero_num	5040.0	2.463294	1.091671	1.0	2.0	2.0	

	75%	max	varianza
Edad	52.0	60.0	128.895626
Años_Expe	26.0	35.0	95.307620
Hrs_semana	49.0	60.0	132.494356
Balance_vida_trabajo	4.0	5.0	1.944816
Calificación_aislamientos	4.0	5.0	1.910463
Apoyo_de_empresa	4.0	5.0	1.868971
Genero_num	3.0	4.0	1.191746

Resumen de variables categoricas

Resumen de variables categoricas

Genero:

Genero

Male 1493

Female 1196

Prefer not to say 1188

Non-binary 1171

Name: count, dtype: int64

Rol:

Rol

Project Manager 987

Sales 789

HR 687

Software Engineer 677

Data Scientist 678

Designer 669

Marketing 641

Name: count, dtype: int64

Industria:

Industria

Retail 996

Healthcare 782

IT 699

Finance 692

Manufacturing 678

Education 641

Consulting 648

Name: count, dtype: int64

Ubicacion:

Ubicacion

Remote 1894

Hybrid 1628

Onsite 1526

Name: count, dtype: int64

Nivel_estres:

Nivel_estres

High 1985

Medium 1571

Low 1564

Name: count, dtype: int64

Salud_Mental:

Salud_Mental

Anxiety 2659

Burnout 1228

Depression 1161

Name: count, dtype: int64

Recurso_SaludM:

Recurso_SaludM

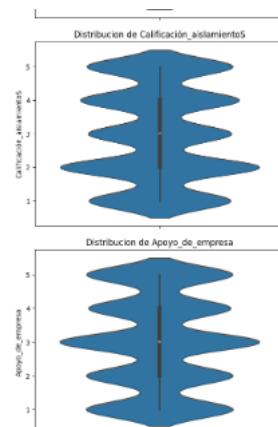
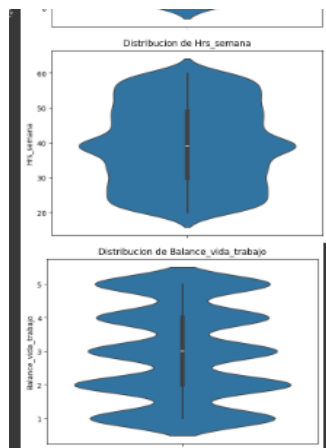
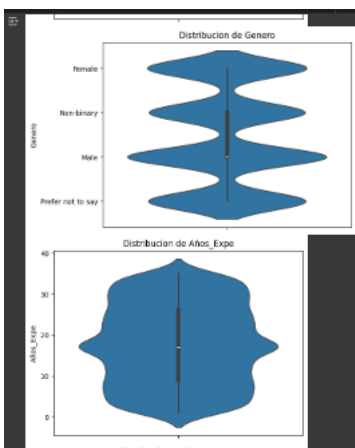
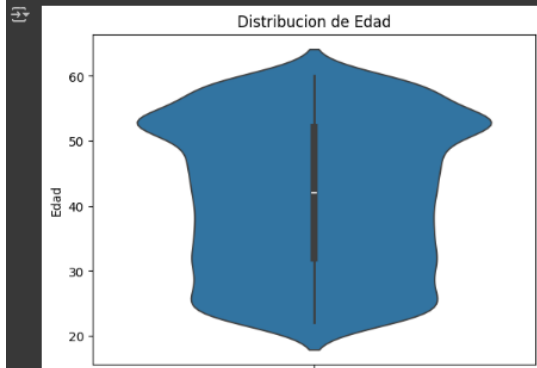
No 2696

Visualización

Graficos de variables numericas

```
364] #variables numericas
#graficas de violines
col=['Edad','Genero','Años_Expe','Hrs_semana','Balance_vida_trabajo', 'Calificación_aislamientos', 'Apoyo_de_empresa']
for i in col:
    sns.violinplot(y=i,data=df)

    plt.title(f'Distribucion de {i}')
    plt.show()
```



Descripción. En este análisis podemos observar:

- que la media ronda entre los 30 años, siendo personas jóvenes quienes abundan en estas modalidades
- la categoría "femenino" tiene la proporción más alta, seguida de "masculino". Las categorías "no binario" y "prefiero no decir" tienen proporciones más pequeñas.
- De igual manera muestra que hay un mayor número de trabajadores con menos años de experiencia, con un conjunto de 20 años o menos.
- Las horas trabajadas por semana son alrededor de 40 horas o menos sin embargo hay una variabilidad considerable de algunas personas trabajando muchas menos horas y otras muchas mas
- en el balance de vida hay una ligera tendencia reportando un balance similar de cada persona, siendo que la mayoría considera un balance moderadamente bueno

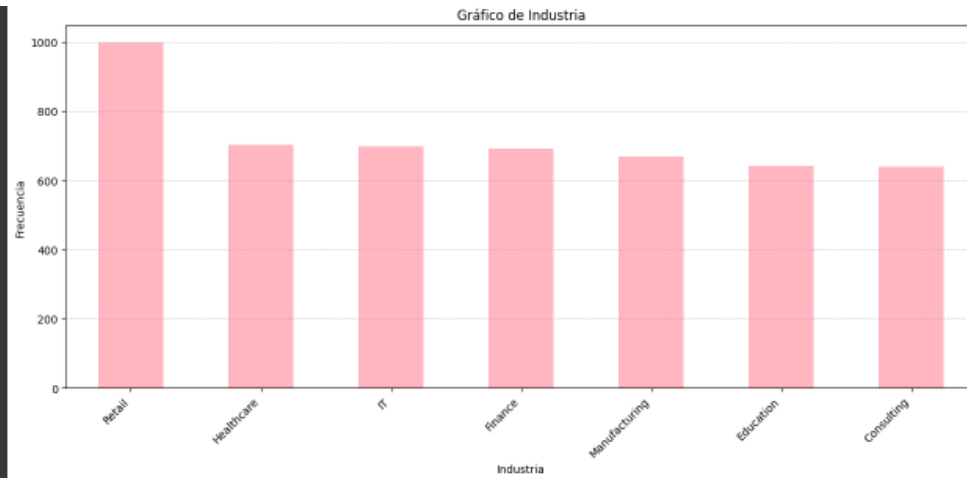
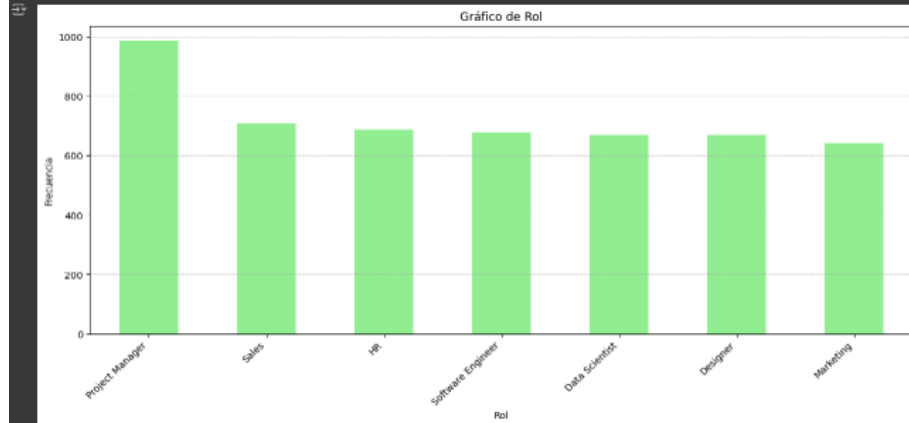
Gráficos de variables categoricas

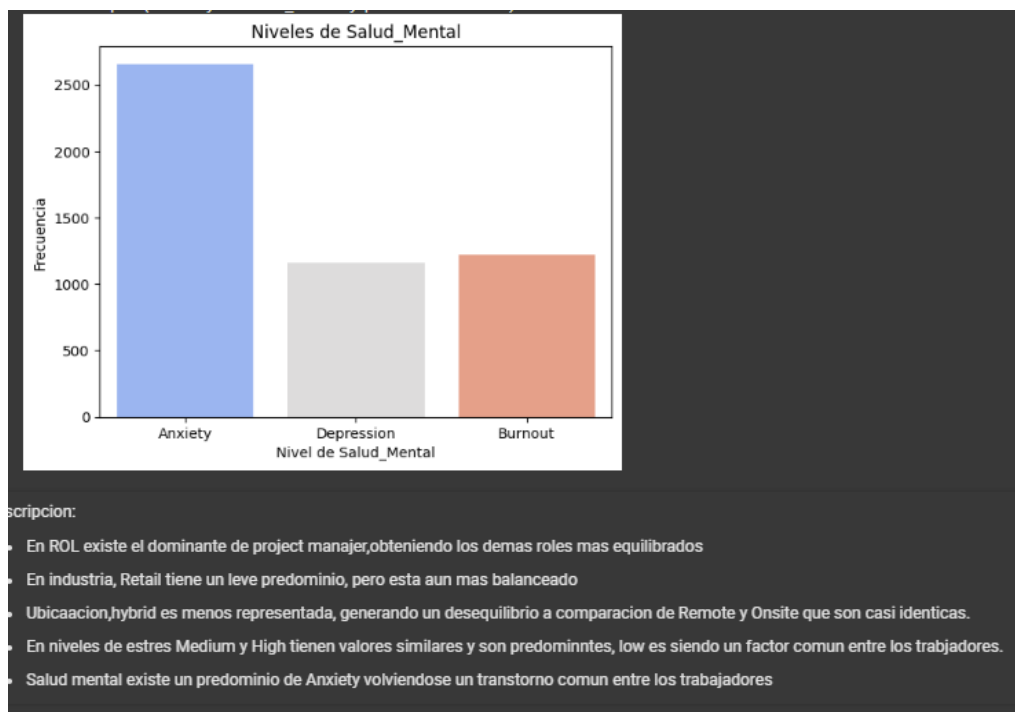
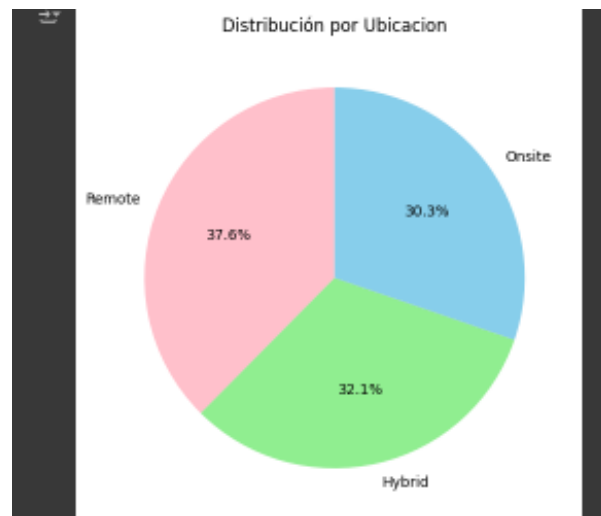
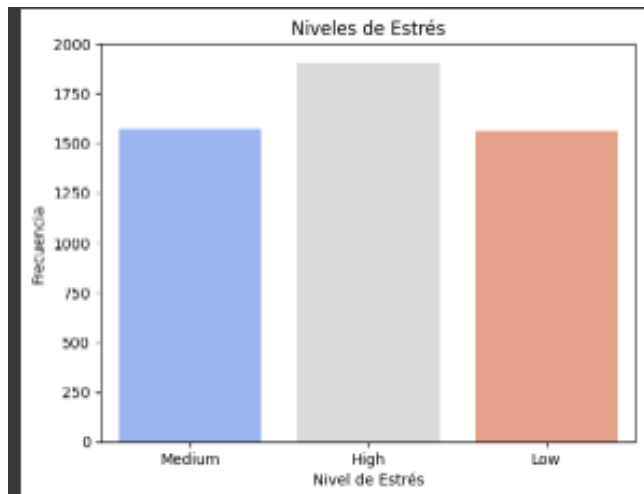
```
201] #Gráfico de rol
plt.figure(figsize=(15, 6))
df['rol'].value_counts().plot(kind='bar', color='lightgreen')
plt.title('Gráfico de rol')
plt.xlabel('rol')
plt.ylabel('frecuencia')

# Rotar las etiquetas del eje X si es necesario
plt.xticks(rotation=45, ha='right')

# Añadir rejilla opcional
plt.grid(True, which='both', axis='y', linestyle='--', linewidth=0.7, alpha=0.7)

# Mostrar el gráfico
plt.show()
```





Correlacion entre variables

Cambio de la tipos de variables , de texto a numerico para poder observar si tiene algun tipo de influencia en el analisis, y poder visualizirlo en el mapa de calor

```
df2=df.copy()
# Asignar valores numericos a cada categoria de 'Genero'
df2['Genero'] = df2['Genero'].map({'Female': 1, 'Male': 2, 'Non-binary':3, 'Prefer not to say':4})
df2['Recurso_SaludM'] = df2['Recurso_SaludM'].map({'No': 1, 'Yes': 2})
df2['Ubicacion'] = df2['Ubicacion'].map({'Remote':1, 'Hybrid':2, 'Onsite':3})
df2
```

	Edad	Genero	Id	Industria	Años_Expe	Ubicacion	Hrs_semana	Balance_vida_trabajo	Nivel_estres	Salud_Mental	Recurso_saludM	Cambio_en_productividad	Calificación_aislamientos	Satisfacción_trabajo	Apoyo_de_empresa	Calidad_del_sueño	Region
0	40.0	1	Data Scientist	IT	3.0	1	52.0	1.0	Medium	Anxiety	1	Increase	3.0	Satisfied	2.0	Good	Asia
1	59.0	3	Software Engineer	Education	22.0	2	46.0	5.0	Medium	Anxiety	1	No Change	4.0	Unsatisfied	5.0	Poor	North America
2	27.0	2	Software Engineer	Finance	20.0	3	32.0	4.0	High	Depression	2	Increase	3.0	Unsatisfied	3.0	Poor	Europe
3	49.0	2	Sales	Consulting	32.0	3	35.0	2.0	High	Anxiety	2	Decrease	3.0	Unsatisfied	3.0	Average	North America
4	59.0	3	Sales	IT	31.0	2	39.0	4.0	High	Anxiety	1	Increase	5.0	Unsatisfied	1.0	Average	South America
...
5035	56.0	4	Software Engineer	Education	5.0	2	35.0	5.0	Medium	Anxiety	2	Increase	2.0	Neutral	5.0	Average	Asia
5036	41.0	4	Project Manager	IT	9.0	3	50.0	2.0	High	Anxiety	2	No Change	5.0	Neutral	5.0	Average	Africa
5037	28.0	1	Marketing	Retail	6.0	3	52.0	4.0	Low	Burnout	2	Increase	1.0	Unsatisfied	1.0	Poor	Asia
5038	58.0	2	Data Scientist	Education	19.0	1	34.0	2.0	Medium	Anxiety	1	Decrease	5.0	Satisfied	3.0	Good	North America
5039	59.0	3	Marketing	Manufacturing	20.0	3	21.0	3.0	High	Burnout	1	Increase	1.0	Neutral	1.0	Average	Africa

Próximos pasos: Generar código con df2 Ver gráficos recomendados New interactive sheet

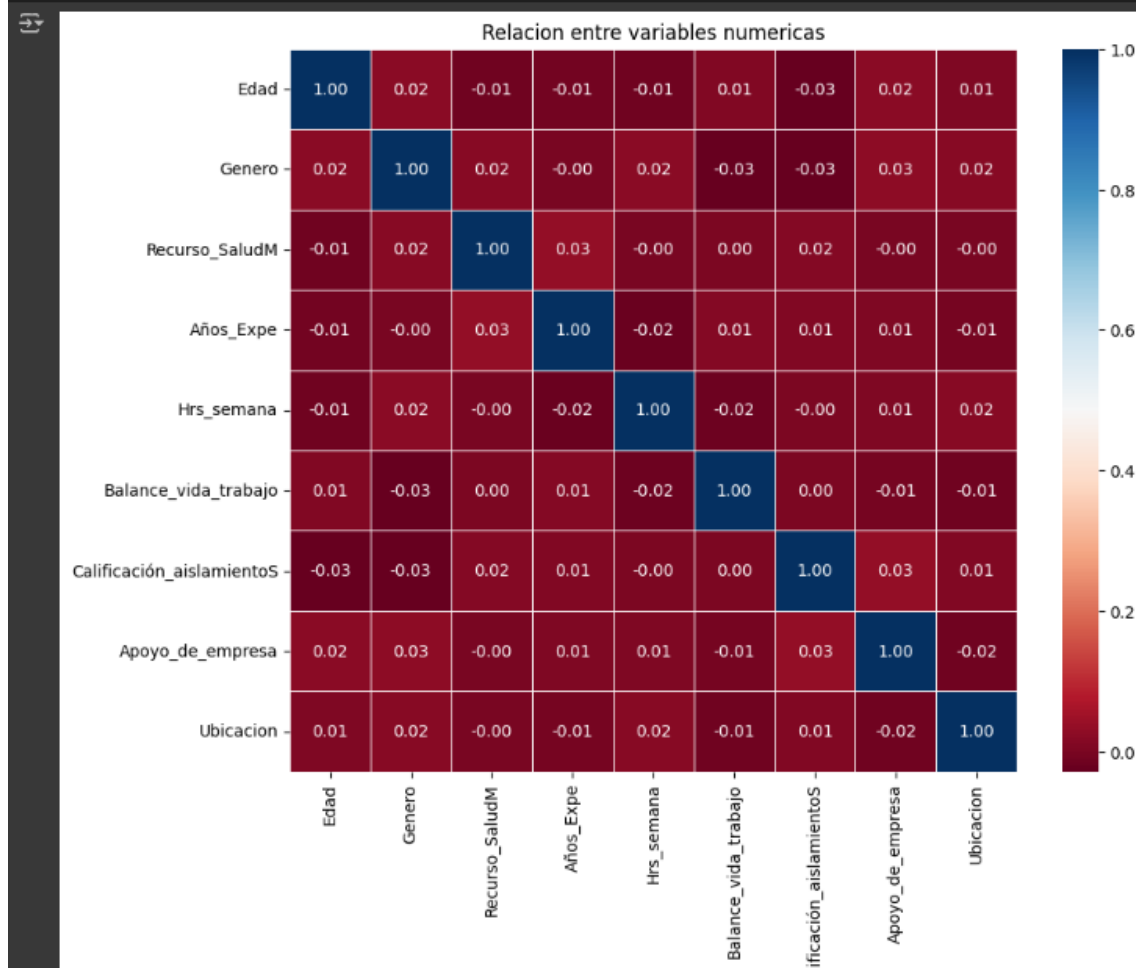
```
[30] # Matriz de correlacion
df3= df2.copy()[['Edad','Genero','Recurso_SaludM','Años_Expe','Hrs_semana','Balance_vida_trabajo', 'calificación_aislamientos', 'Apoyo_de_empresa','Ubicacion'],]
# Calcular la matriz de correlación
cm1 = df3.corr()
cm1
```

	Edad	Genero	Recurso_SaludM	Años_Expe	Hrs_semana	Balance_vida_trabajo	Calificación_aislamientos	Apoyo_de_empresa	Ubicacion
Edad	1.000000	0.020205	-0.014047	-0.010886	-0.013769	0.008690	-0.028786	0.022821	0.006059
Genero	0.020205	1.000000	0.018600	-0.001649	0.019960	-0.025943	-0.025772	0.027217	0.016281
Recurso_SaludM	-0.014047	0.018600	1.000000	0.032883	-0.004077	0.001025	0.015085	-0.000657	-0.002837
Años_Expe	-0.010886	-0.001649	0.032883	1.000000	-0.022774	0.011455	0.010442	0.005815	-0.006851
Hrs_semana	-0.013769	0.019960	-0.004077	-0.022774	1.000000	-0.019457	-0.000511	0.006222	0.016906
Balance_vida_trabajo	0.008690	-0.025943	0.001025	0.011455	-0.019457	1.000000	0.000439	-0.008060	-0.014896
Calificación_aislamientos	-0.028786	-0.025772	0.015085	0.010442	-0.000511	0.000439	1.000000	0.031565	0.008430
Apoyo_de_empresa	0.022821	0.027217	-0.000657	0.005815	0.006222	-0.008060	0.031565	1.000000	-0.015678
Ubicacion	0.006059	0.016281	-0.002837	-0.006851	0.016906	-0.014896	0.008430	-0.015678	1.000000

Próximos pasos: Generar código con cm1 Ver gráficos recomendados New interactive sheet

- Edad se muestra con una relacion debil con el resto de variables, no es un factor determinante, pero con los años de experiencia tiene una logica relacion ya que a mayor edad se genera mas experiencia
- Horas semanales y balance vida-trabajo, ligera relacion , mas horas trabajadas suele implicar mejor tiempo para la vida personal.
- Calificación de aislamiento y apoyo de la empresa, las personas que se sienten mas aisladas podrian percibir menos apoyo de la empresa.

```
[31] #Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(cm1, annot=True, cmap='RdBu', fmt=".2f", linewidths=0.5)
plt.title('Relacion entre variables numericas')
plt.show()
```



✓ Analisis de valores atipicos (outliers)

```
[33] numericas = df3.select_dtypes(include='number')

# Detectar outliers por IQR
outliers = {}
for col in numericas.columns:
    Q1 = numericas[col].quantile(0.25)
    Q3 = numericas[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 2 * IQR
    upper_bound = Q3 + 2 * IQR
    outliers[col] = numericas[(numericas[col] < lower_bound) | (numericas[col] > upper_bound)][col]

# Mostrar columnas con outliers detectados
for col, vals in outliers.items():
    print(f"Outliers detectados en '{col}': {len(vals)} valores")
```

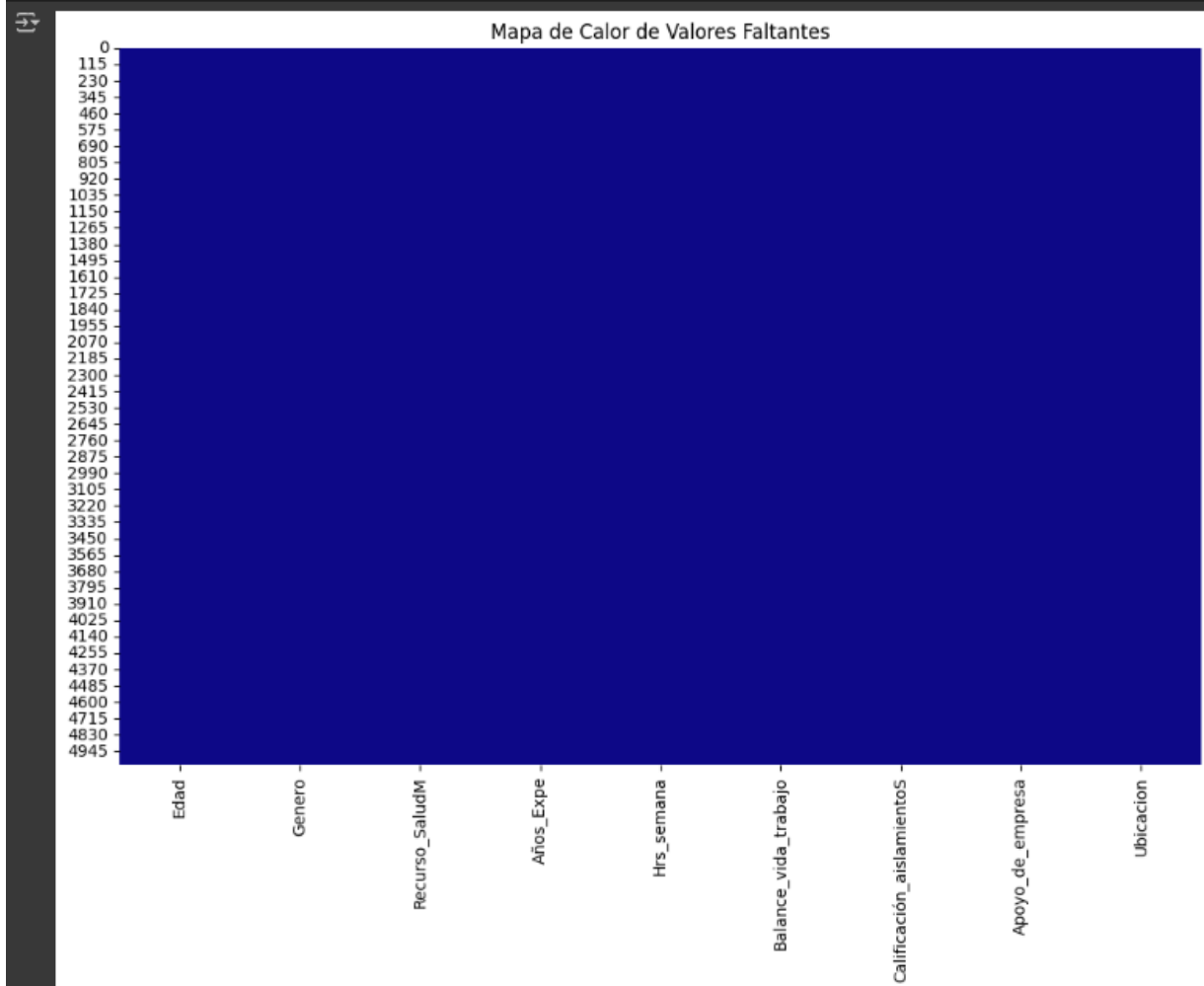
```
Outliers detectados en 'Edad': 0 valores
Outliers detectados en 'Genero': 0 valores
Outliers detectados en 'Recurso_SaludM': 0 valores
Outliers detectados en 'Años_Expe': 0 valores
Outliers detectados en 'Hrs_semana': 0 valores
Outliers detectados en 'Balance_vida_trabajo': 0 valores
Outliers detectados en 'Calificación_aislamientos': 0 valores
Outliers detectados en 'Apoyo_de_empresa': 0 valores
Outliers detectados en 'Ubicacion': 0 valores
```

Sin datos atipicos encontrados.

```
[34] #visualizar datos faltantes
faltantes = df3.isnull().mean() * 100
print("Porcentaje de valores faltantes por columna:")
print(faltantes)
```

```
Porcentaje de valores faltantes por columna:
Edad                0.0
Genero              0.0
Recurso_SaludM     0.0
Años_Expe          0.0
Hrs_semana         0.0
Balance_vida_trabajo 0.0
Calificación_aislamientos 0.0
Apoyo_de_empresa   0.0
Ubicacion          0.0
dtype: float64
```

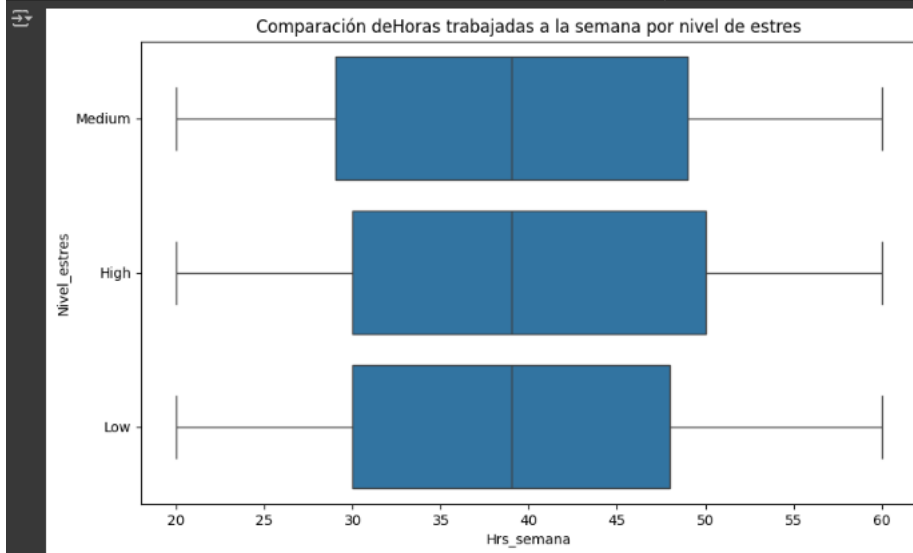
```
[35] plt.figure(figsize=(12, 8))
sns.heatmap(df3.isnull(), cbar=False, cmap='plasma')
plt.title("Mapa de Calor de Valores Faltantes")
plt.show()
```



Como se noto en la tabla y en el mapa de calor, no existen datos faltantes

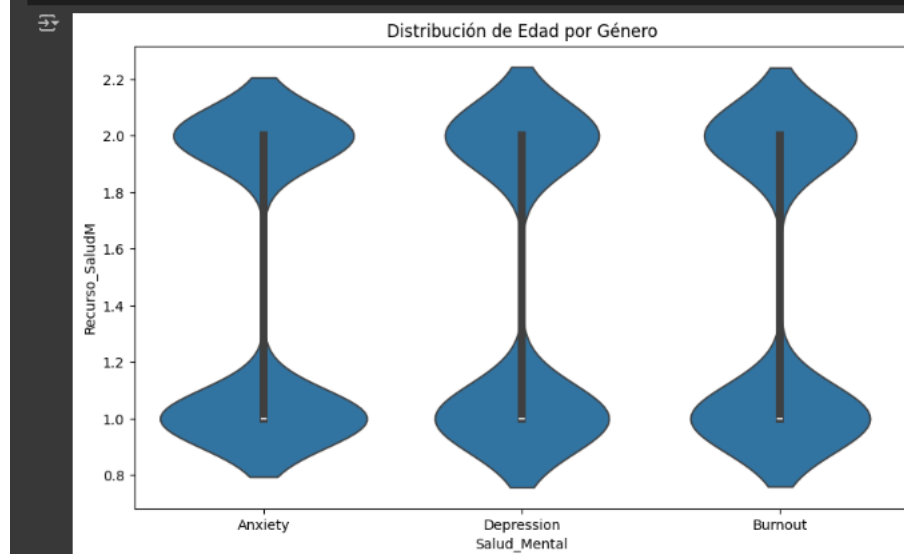
Relacion entre variables categoricas y numericas

```
[36] plt.figure(figsize=(10, 6))
sns.boxplot(x='Hrs_semana', y='Nivel_estres', data=df2.copy())
plt.title("Comparación de Horas trabajadas a la semana por nivel de estrés")
plt.show()
```



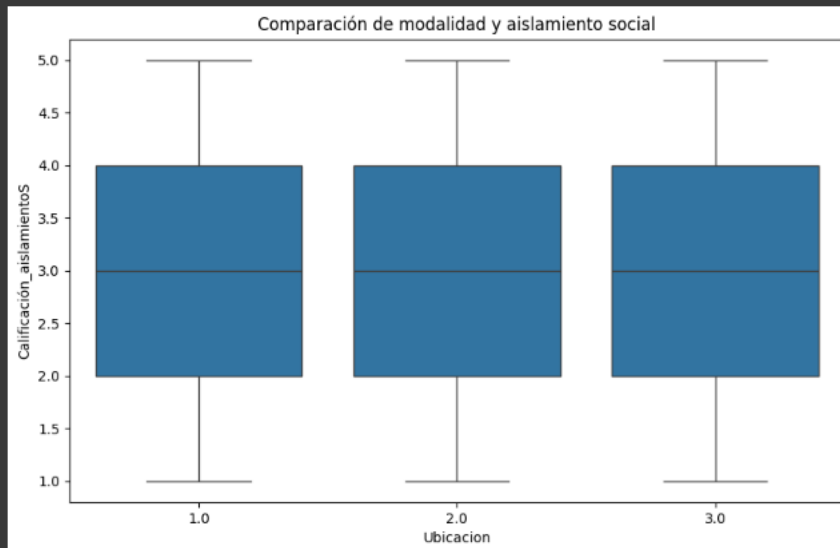
La mediana de las horas trabajadas parece ser ligeramente más alta para el nivel de estrés medio, en comparación con los niveles bajo y alto. Esto indicaría que, en promedio, las personas que reportan un nivel de estrés medio trabajan más horas a la semana.

```
[37] plt.figure(figsize=(10, 6))
sns.violinplot(x='Salud_Mental', y='Recurso_SaludM', data=df2.copy())
plt.title("Distribución de Edad por Género")
plt.show()
```



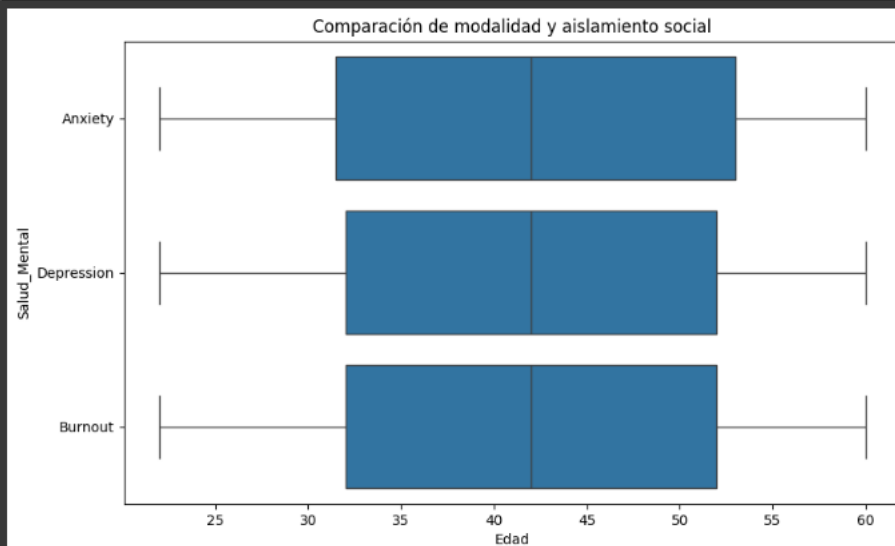
Los tres grupos (ansiedad, depresión y burnout) parecen tener una distribución similar sugiere que la concentración de los datos es similar en cada grupo aunque podría haber ligeras variaciones que no son tan evidentes a simple vista

```
39] plt.figure(figsize=(10, 6))
sns.boxplot(x='Ubicacion', y='Calificación_aislamientos', data=df2.copy())
plt.title("Comparación de modalidad y aislamiento social")
plt.show()
```



tienen una longitud similar y las medianas se encuentran aproximadamente en el mismo punto a lo largo del eje Y. Esto indica que la distribución de la calificación de aislamiento social es bastante similar en las tres modalidades.

```
plt.figure(figsize=(10, 6))
sns.boxplot(x='Edad', y='Salud_Mental', data=df2.copy())
plt.title("Comparación de modalidad y aislamiento social")
plt.show()
```



Podemos notar que no hay mucha diferencia entre los análisis de variables categoricas y numericas, tomemos en cuenta en cuenta que la mayoría de variables tienen datos repetidos, ya que al analizar comportamientos similares, las respuestas suelen ser las mismas, apesar de que los transtornos no siempre se comportan igual en cada individuo, podemos notar que en el ambito laboral se esta mostrando de manera casi identica.

Observaciones.

En el análisis de Eda pudimos notar que las variables tienen datos muy similares, los datos son repetitivos, (como la variable de ubicación, donde solo son 3 modalidades presentes, pasa lo mismo con casi todas las variables).

Nuestra meta es saber cómo la modalidad de trabajo afecta a la salud mental de los trabajadores con ayuda de otros factores, los años de experiencia, las horas trabajadas; considero que al ser un tema de salud es importante que analicemos con mayor detención los datos, al tomar una mayor cantidad de datos podremos saber estrategias para ayudar, dejando en claro que la variable de ubicación (modalidad de trabajo) es fundamental para reflejar el patrón significativo de los datos. La variable podría llegar a ser un predictor natural en el modelo que realizaremos más adelante.

Existen patrones muy obvios que ayudan a que el entendimiento de las correlaciones sea más fácil de entender como lo son los años de experiencia con la edad.

Al tener variables con poca correlación es complicado analizar numéricamente los datos, sin embargo, más adelante realizaremos un “cambio” en el tipo de dato, para poder continuar con el análisis.

En la interpretación de los futuros hallazgos, podríamos tener cierta incertidumbre, ya que la mayoría de nuestras variables son de tipo categóricas, causando un poco de preocupación. Anteriormente se mencionó el cambio de dato, pero al finalizar el modelo tendremos que prestar mayor atención a el entendimiento de los resultados y poder obtener información segura y firme.

Modelo de Machine Learning.

Modelo elegido: "Random Forest Classifier"

Justificación: El modelo puede identificar cuales variables tienen mayor peso en la predicción de la salud mental en base a la modalidad de trabajo. Esto permitirá si realmente la ubicación tiene algun impacto significativo con las demás variables ¿.

Es mas resistente al sobreajuste, especialmente por la gran cantidad de números, ya que, al analizar por región, la cantidad puede crecer en un futuro. El objetivo es poder brindar ayuda a aquellas personas que trabajen a distancia, ellos estando en su mayoría en lugares no fijo hace que el registro de cada uno de ellos llegue a ser muy robusto.

Mas adelante intente complementar con una regresión lineal, como método explicativo y tener conclusiones más interpretables.

En un futuro estos modelos podrían ir acompañados de alguna otra técnica como la cauterización, que más adelante detallare. Por el momento solo es una mención, los datos categóricos son los que hacen que se pueda llegar a tomar esta decisión.

”Random Forest Classifier”

Al realizar el cambio de variable a numerico, tome la decision de eliminar ciertas columnas para facilitar el trabajo del analisis y poder ser mas concreto con las variables que me podrian ayudar al abordar las estrategias

```
dfs.drop(columns=['cambio_en_productividad', 'satisfacción_trabajo', 'Region', ], inplace=True)
```

	Edad	Genero	Rol	Industria	Años_Expe	Ubicacion	Hrs_semana	Balance_vida_trabajo	Nivel_estres	Salud_Mental	Recurso_SaludM	Calificación_aislamientos	Apoyo_de_empresa	Calidad_del_sueño	
0	40.0	1.0	1	1	3.0	1.0	52.0	1.0	2	1	1.0	3.0	2.0	1	
1	59.0	3.0	2	2	22.0	2.0	46.0	5.0	2	1	1.0	4.0	5.0	2	
2	27.0	2.0	2	3	20.0	3.0	32.0	4.0	3	2	2.0	3.0	3.0	2	
3	49.0	2.0	3	4	32.0	3.0	35.0	2.0	3	1	2.0	3.0	3.0	3	
4	59.0	3.0	3	1	31.0	2.0	39.0	4.0	3	1	1.0	5.0	1.0	3	
...	
5035	56.0	4.0	2	2	5.0	2.0	35.0	5.0	2	1	2.0	2.0	5.0	3	
5036	41.0	4.0	6	1	9.0	3.0	50.0	2.0	3	1	2.0	5.0	5.0	3	
5037	28.0	1.0	4	7	5.0	3.0	52.0	4.0	1	3	2.0	1.0	1.0	2	
5038	58.0	2.0	1	2	19.0	1.0	34.0	2.0	2	1	1.0	5.0	3.0	1	
5039	59.0	3.0	4	5	20.0	3.0	21.0	3.0	3	3	1.0	1.0	1.0	3	

5040 rows x 14 columns

```
None

[89] from sklearn.model_selection import train_test_split
      from sklearn.preprocessing import LabelEncoder

      # Dividir en características (X) y objetivo (y)
      X = dfs.drop('Ubicacion', axis=1)
      y = dfs['Salud_Mental']

      # Dividir en conjuntos de entrenamiento y prueba
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

La variable x contiene todas las columnas excepto la columna de ubicacion, para poder predecir la variable objetivo. Por otro lado, la variable Y es aquella que se quiere predecir, es el objetivo.

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42): 1 Divide los datos en cuatro conjuntos: X_train: Características del conjunto de entrenamiento. X_test: Características del conjunto de prueba. y_train: Valores de la variable objetivo del conjunto de entrenamiento. y_test: Valores de la variable objetivo del conjunto de prueba. test_size=0.3: Indica que el 30% de los datos se utilizará para el conjunto de prueba y el 70% restante para el conjunto de entrenamiento. random_state=42: Fija una semilla aleatoria para garantizar que la división de los datos sea reproducible.

```
[90] from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix

# Entrenar un modelo Random Forest
rf = RandomForestClassifier(random_state=42, n_estimators=100, max_depth=10)
rf.fit(X_train, y_train)

# Predicciones
y_pred = rf.predict(X_test)

# Métricas de evaluación
print("Matriz de Confusión:")
print(confusion_matrix(y_test, y_pred))

print("\nReporte de Clasificación:")
print(classification_report(y_test, y_pred))
```

```
Matriz de Confusión:
[[791  0  0]
 [ 0 338  0]
 [ 0  0 383]]
```

```
Reporte de Clasificación:
precision    recall  f1-score   support

     1       1.00      1.00      1.00       791
     2       1.00      1.00      1.00       338
     3       1.00      1.00      1.00       383

 accuracy          1.00          1.00          1.00      1512
 macro avg          1.00          1.00          1.00      1512
weighted avg          1.00          1.00          1.00      1512
```

`rf = RandomForestClassifier(random_state=42, n_estimators=100, max_depth=10)`: Crea un objeto de la clase `RandomForestClassifier` con los siguientes parámetros: `random_state=42`: Fija una semilla aleatoria para garantizar que los resultados sean reproducibles. `n_estimators=100`: Especifica el número de árboles en el bosque aleatorio (100 en este caso). `max_depth=10`: Establece la profundidad máxima de cada árbol en el bosque (10 niveles). `rf.fit(X_train, y_train)`: Entrena el modelo de Random Forest utilizando los datos de entrenamiento `X_train` (características) y `y_train` (etiquetas). Predicciones:

`y_pred = rf.predict(X_test)`: Utiliza el modelo entrenado para hacer predicciones sobre el conjunto de prueba `X_test`. Las predicciones se almacenan en la variable `y_pred`. Evaluación del modelo:

el modelo tiene un rendimiento excelente, con una precisión, recall y F1-score de 1.00 para todas las clases. Esto significa que el modelo está clasificando correctamente todos los ejemplos en el conjunto de prueba.

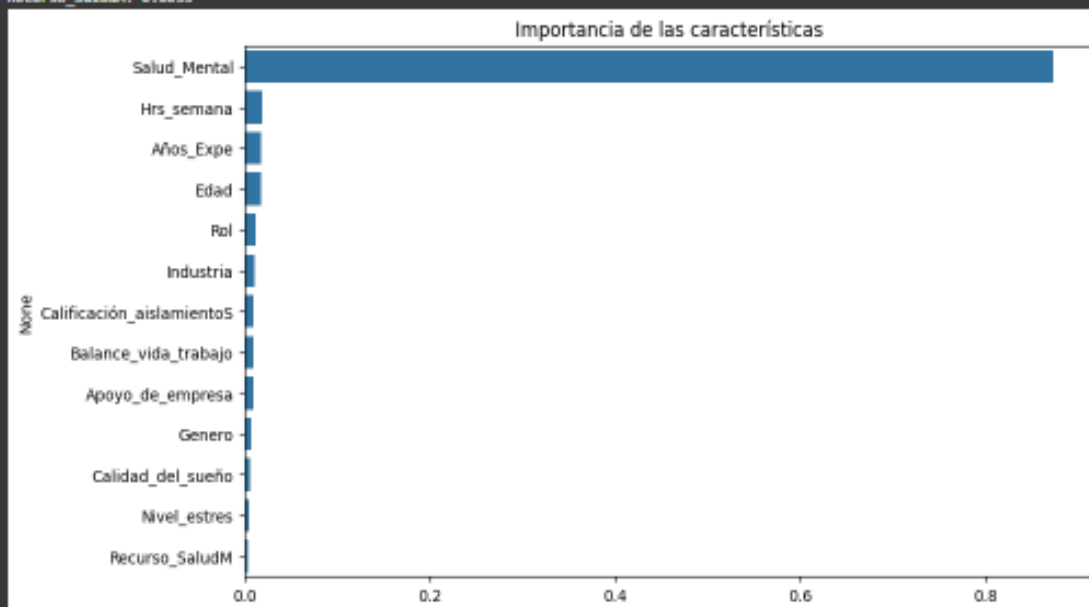
está clasificando correctamente todos los ejemplos en el conjunto de prueba.

```
# Importancia de las características
importances = rf.feature_importances_
features = X.columns
indices = np.argsort(importances)[::-1]

print("Características más importantes:")
for i in indices:
    print(f"{features[i]}: {importances[i]:.4f}")

# Visualización de las importancias
plt.figure(figsize=(10, 6))
sns.barplot(x=importances[indices], y=features[indices])
plt.title("Importancia de las características")
plt.show()
```

```
Características más importantes:
Salud_Mental: 0.8732
Hrs_semana: 0.0192
Años_Expe: 0.0181
Edad: 0.0175
Rol: 0.0117
Industria: 0.0103
Calificación_aislamientoS: 0.0096
Balance_vida_trabajo: 0.0092
Apoyo_de_empresa: 0.0091
Genero: 0.0072
Calidad_del_sueño: 0.0063
Nivel_estres: 0.0053
Recurso_SaludM: 0.0033
```



Obtener las importancias de las características: Utiliza el atributo `feature_importances_` del modelo de Random Forest para obtener un arreglo con la importancia de cada característica. Ordenar las características: Ordena las características de mayor a menor importancia utilizando `np.argsort`.

El modelo de Random Forest considera que la "Salud Mental" es el factor más importante para realizar las predicciones. Otras características como "Hrs_semana", "Años_Expe" y "Edad" también tienen una influencia significativa. Características como "Apoyo de empresa" y "Género" parecen tener una influencia menor en las predicciones.

```
[70] from sklearn.model_selection import GridSearchCV
```

```
[70] # Definir los parámetros para ajustar
param_grid = {
    'n_estimators': [50, 100, 150],
    'max_depth': [5, 10, 20],
    'min_samples_split': [2, 5, 10]
}

# Búsqueda de cuadrícula
grid_search = GridSearchCV(RandomForestClassifier(random_state=42), param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

# Mejor modelo
print("Mejores parámetros:", grid_search.best_params_)
best_model = grid_search.best_estimator_

# Evaluación con el mejor modelo
y_pred_best = best_model.predict(X_test)
print("\nReporte de Clasificación con Modelo Optimizado:")
print(classification_report(y_test, y_pred_best))
```

Mejores parámetros: {'max_depth': 5, 'min_samples_split': 2, 'n_estimators': 50}

Reporte de Clasificación con Modelo Optimizado:

	precision	recall	f1-score	support
1	1.00	1.00	1.00	791
2	1.00	1.00	1.00	338
3	1.00	1.00	1.00	383
accuracy			1.00	1512
macro avg	1.00	1.00	1.00	1512
weighted avg	1.00	1.00	1.00	1512

Una vez encontrado el mejor conjunto de parámetros, se evalúa el modelo en un conjunto de datos de prueba y se presentan los resultados.

Ajuste de hiperparámetros: Se busca la mejor configuración para el modelo. Validación cruzada: Se evalúa el modelo de forma más robusta para evitar el sobreajuste. Mejores parámetros: Se identifica la mejor combinación de parámetros. Evaluación: Se mide el rendimiento del modelo en un conjunto de datos independiente.

Precisión, Recall y F1-score de 1.00 para todas las clases: Esto indica que el modelo ha clasificado correctamente todos los ejemplos en el conjunto de prueba. Es decir, el modelo es capaz de distinguir perfectamente entre las diferentes clases. En este caso, vemos que hay un desbalance en las clases, ya que la clase 1 tiene más ejemplos que las otras. Sin embargo, el modelo ha logrado obtener rendimiento perfecto incluso en este escenario. Métricas promedio: Tanto el promedio ponderado como el promedio macro de las métricas también son de 1.00, lo que refuerza la idea de que el modelo tiene un desempeño excepcional. Interpretación:

Modelo altamente preciso: El modelo de Random Forest, con los parámetros ajustados mediante búsqueda por cuadrícula, ha logrado un rendimiento perfecto en la tarea de clasificación.

```
[94] from sklearn.model_selection import cross_val_score
scores = cross_val_score(rf, X, y, cv=5)
print("Puntajes de validación cruzada:", scores)
print("Precisión promedio:", scores.mean())
```

Puntajes de validación cruzada: [1. 1. 1. 1. 1.]
Precisión promedio: 1.0

Una precisión de 1.0 significa que el modelo ha realizado todas las predicciones correctamente. En otras palabras, el modelo es capaz de distinguir perfectamente entre las diferentes clases presentes en los datos.

REGRESION LINEAL

```
[136] # Calcular el coeficiente de correlación de Pearson
correlacion_pearson = df5['Ubicacion'].corr(df5['Salud_Mental'], method='pearson')

print(f"Coefficiente de correlación de Pearson: {correlacion_pearson}")
```

Coeficiente de correlación de Pearson: 0.020968254651003226

```
X = df5[['Ubicacion']]
y = df5['Salud_Mental']
# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=42)
```

```
[142] # Crear y entrenar el modelo
model = LinearRegression()
model.fit(X_train, y_train)

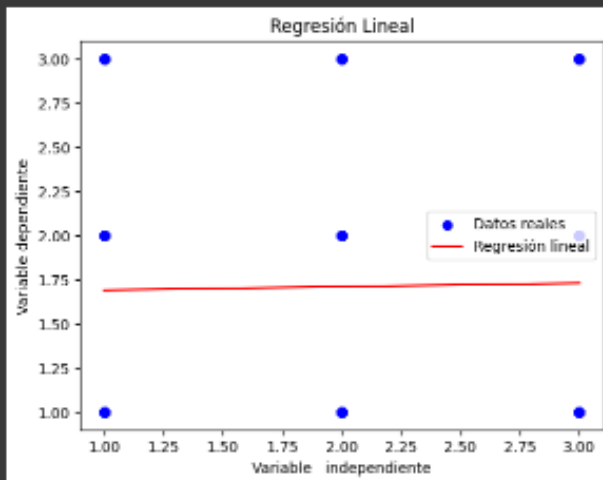
# Hacer predicciones
y_pred = model.predict(X_test)
```

Generar create a dataframe with 2 columns and 10 rows

```
[144] # Evaluar el modelo
mse = mean_squared_error(y_test, y_pred)
print("Error cuadrático medio:", mse)
```

Error cuadrático medio: 0.7115893696663909

```
[145] # Visualizar los resultados
plt.scatter(X_test, y_test, color='blue', label='Datos reales')
plt.plot(X_test, y_pred, color='red', label='Regresión lineal')
plt.xlabel('Variable independiente')
plt.ylabel('Variable dependiente')
plt.legend()
plt.title('Regresión Lineal')
plt.show()
```



¿?

Conclusiones y Futuras Líneas de Trabajo

En el análisis vimos como las variables de ubicación tienen relación con la de salud mental, damos por entendido que la modalidad en la que se trabaja puede llegar a afectar ese ámbito de la vida tan importante, tomamos en cuenta que otras variables o mejor dicho factores de la vida personal pueden influir en el sentir de cada trabajador. Al darnos cuenta de que componentes son los perjudicados, se comienza a personalizar las estrategias de atención de acuerdo con las necesidades de los grupos.

En un futuro podría manejar una técnica de clusterización, de esa manera podríamos manejar las variables de manera categórica y poder separar conjuntos similares, ajustando los tratamientos, conferencias o atención personalizada; facilitando la identificación de los problemas mentales.

Esa técnica ayuda a que la creación de espacios seguros en el trabajo sea eficaz. Incluso poder disminuir el porcentaje de empleados que son diagnosticados con algún trastorno, pues al comenzar presentando características similares de algún grupo identificado con problemas mentales, poder abordar y evitar que avance la problemática.

Después de analizar correctamente las variables, podríamos tomar muchos más factores y ya no solo centrarnos en la modalidad de trabajo, sino más en el tipo de profesión, ambientes laborales e incluso hasta como se desenvuelven los trabajadores en situaciones extremas y poder desarrollar técnicas para poder amenizar esas presiones, buscando que a la hora de trabajar no se desarrollen todos entornos conflictivos con su persona o con algún conjunto mayor.