

## ▼ Mastografías en México de 2013 a 2023

```
from google.colab import drive
drive.mount('/content/drive')
```

→ Mounted at /content/drive

### › Mastografías en mexico

[ ] ↴ 56 celdas ocultas

## ▼ tasa de mortalidad en mexico

```
import pandas as pd
```

```
file_path = '/content/drive/MyDrive/Cancer de mama/Tasa de mortalidad de 1990 - 2013 en mexico.csv'
df = pd.read_csv(file_path)
df
```

→

	Periodo	Entidad de Residencia	Defunciones por Cancer de Mama	Poblacion Femenina Total	Tasa de Mortalidad por Cancer de Mama
0	1990	Aguascalientes	29	159967	18.128739
1	1990	Baja California	79	337917	23.378522
2	1990	Baja California Sur	7	63290	11.060199
3	1990	Campeche	12	108154	11.095290
4	1990	Coahuila de Zaragoza	62	410228	15.113547
...	...	...	...	...	...
787	2013	Tlaxcala	45	340359	13.221334
788	2013	Veracruz Llave	353	2304554	15.317497
789	2013	Yucatán	80	584269	13.692323
790	2013	Zacatecas	82	424179	19.331461

```
import pandas as pd
```

```
# Suponiendo que ya tienes un DataFrame llamado df
# Agrupar por "Periodo" y sumar las columnas relevantes
df_grouped = df.groupby('Periodo').agg({
    'Defunciones por Cancer de Mama': 'sum',
    'Poblacion Femenina Total': 'sum'
}).reset_index()
```

```
# Calcular la tasa de mortalidad
df_grouped['Tasa de Mortalidad por Cancer de Mama'] = (df_grouped['Defunciones por Cancer de Mama'] / df_grouped['Poblacion Femenina Total']) * 100000
```

```
# Ahora df_grouped tiene los resultados agregados por año y la tasa recalculada
df_grouped
```



Periodo	Defunciones por Cancer de Mama	Poblacion Femenina Total	Tasa de Mortalidad por Cancer de Mama
0	4380	35852664	12.216665
1	4668	37007840	12.613544
2	5036	38193064	13.185640
3	5352	39392350	13.586394
4	5478	40623620	13.484766
5	5960	41890024	14.227731
6	6130	43182190	14.195667
7	6362	44490494	14.299684
8	6692	45816222	14.606180
9	6764	47132266	14.351103
10	6838	48432714	14.118556
11	7126	49756872	14.321640
12	7644	51131594	14.949661
13	7722	52529180	14.700401
14	8300	53906692	15.396975
15	8410	55250834	15.221490
16	8880	56601780	15.688553
17	9162	57999644	15.796649
18	9604	59435352	16.158733
19	9786	60882268	16.073646
20	10068	62304788	16.159272
21	10412	63710038	16.342794
22	11166	65116806	17.147647
23	10810	66521320	16.250429

```
file_path_2 = '/content/drive/MyDrive/Cancer de mama/Tasa de mortalidad de 1990 - 2013 en m€
df2 = pd.read_csv(file_path_2)
df2
```

→

	AÑO	CLAVE ENTIDAD	ENTIDAD	CLAVE SEXO	SEXO	DEFUNCIONES POR CÁNCER DE MAMA EN MUJERES DE 25 AÑOS O MÁS	POBLACIÓN DE MUJERES DE 25 AÑOS O MÁS	TAZA DE MORTALIDAD POR CÁNCER DE MAMA
0	2010	0	Nacional	2	Mujeres	5037	31152392	16.17
1	2011	0	Nacional	2	Mujeres	5207	31855025	16.35
2	2012	0	Nacional	2	Mujeres	5584	32558390	17.15
3	2013	0	Nacional	2	Mujeres	5526	33260659	16.61
4	2014	0	Nacional	2	Mujeres	5974	33967702	17.59
...	...	...	...	...	...	...	...	...
226	2012	32	Zacatecas	2	Mujeres	75	416111	18.02
227	2013	32	Zacatecas	2	Mujeres	82	424179	19.33
228	2014	32	Zacatecas	2	Mujeres	78	432332	18.04
229	2015	32	Zacatecas	2	Mujeres	88	440543	19.98
230	2016	32	Zacatecas	2	Mujeres	82	448723	18.27

```
# Ver cuántas respuestas hay para la columna 'SEXO' y el conteo de cada una
sexo_counts = df2['SEXO'].value_counts()
```

```
# Mostrar el resultado
print(sexo_counts)
```

→ SEXO  
 Mujeres 231  
 Name: count, dtype: int64

```
# Obtener los nombres de todas las columnas
columnas = df2.columns
```

```
# Mostrar los nombres de las columnas
print(columnas)
```

→ Index(['AÑO', 'CLAVE ENTIDAD', 'ENTIDAD', 'CLAVE SEXO', 'SEXO',
 'DEFUNCIONES POR CÁNCER DE MAMA EN MUJERES DE 25 AÑOS O MÁS ',
 'POBLACIÓN DE MUJERES DE 25 AÑOS O MÁS',
 'TASA DE MORTALIDAD POR CÁNCER DE MAMA'],
 dtype='object')

```
# Agrupar por "AÑO" y sumar las columnas relevantes
df_grouped2 = df2.groupby('AÑO').agg({
```

```
'DEFUNCIONES POR CÁNCER DE MAMA EN MUJERES DE 25 AÑOS O MÁS': 'sum',
'POBLACIÓN DE MUJERES DE 25 AÑOS O MÁS': 'sum'
}).reset_index()

# Calcular la tasa de mortalidad
df_grouped2['Tasa de Mortalidad por Cancer de Mama'] = (df_grouped2['DEFUNCIONES POR CÁNCER DE MAMA EN MUJERES DE 25 AÑOS O MÁS'] / df_grouped2['POBLACIÓN DE MUJERES DE 25 AÑOS O MÁS']) * 1000000

# Ahora df_grouped2 tiene los resultados agregados por año y la tasa recalculada
df_grouped2
```

	AÑO	DEFUNCIONES POR CÁNCER DE MAMA EN MUJERES DE 25 AÑOS O MÁS	POBLACIÓN DE MUJERES DE 25 AÑOS O MÁS	Tasa de Mortalidad por Cancer de Mama
0	2010	10074	62304782	16.168903
1	2011	10414	63710053	16.345929
2	2012	11168	65116779	17.150725
3	2013	11052	66521317	16.614223
4	2014	11948	67935405	17.587295
5	2015	12552	69358429	18.097296
6	2016	13260	70783734	18.733117

```
# Nuevos nombres de las columnas
nuevos_nombres = ['Año', 'Defunciones por cáncer de mama en mujeres de 25 años o más',
                   'Población de mujeres de 25 años o más', 'Tasa de mortalidad por cáncer de mama']
```

```
# Asignar los nuevos nombres de columna al DataFrame
df_grouped2.columns = nuevos_nombres
```

```
# Verificar los cambios
print(df_grouped2.columns)
```

→ Index(['Año', 'Defunciones por cáncer de mama en mujeres de 25 años o más',  
 'Población de mujeres de 25 años o más',  
 'Tasa de mortalidad por cáncer de mama'],  
 dtype='object')

```
# Asignar los nuevos nombres de columna al DataFrame
df_grouped2.columns = nuevos_nombres
```

```
# Verificar los cambios
print(df_grouped2.columns)
```

→ Index(['Año', 'Defunciones por cáncer de mama en mujeres de 25 años o más',  
 'Población de mujeres de 25 años o más',  
 'Tasa de mortalidad por cáncer de mama'],

```
dtype='object')
```

```
df_grouped
```

	Año	Defunciones por cáncer de mama en mujeres de 25 años o más	Población de mujeres de 25 años o más	Tasa de mortalidad por cáncer de mama
0	1990	4380	35852664	12.216665
1	1991	4668	37007840	12.613544
2	1992	5036	38193064	13.185640
3	1993	5352	39392350	13.586394
4	1994	5478	40623620	13.484766
5	1995	5960	41890024	14.227731
6	1996	6130	43182190	14.195667
7	1997	6362	44490494	14.299684
8	1998	6692	45816222	14.606180
9	1999	6764	47132266	14.351103
10	2000	6838	48432714	14.118556
11	2001	7126	49756872	14.321640
12	2002	7644	51131594	14.949661
13	2003	7722	52529180	14.700401
14	2004	8300	53906692	15.396975
15	2005	8410	55250834	15.221490
16	2006	8880	56601780	15.688553
17	2007	9162	57999644	15.796649
18	2008	9604	59435352	16.158733
19	2009	9786	60882268	16.073646
20	2010	10068	62304788	16.159272
21	2011	10412	63710038	16.342794
22	2012	11166	65116806	17.147647

```
# Filtrar el DataFrame para mantener solo las filas donde 'Año' sea menor que 2010
df_filtrado = df_grouped.loc[df_grouped['Año'] < 2010]
```

```
# Ahora df_filtrado tiene solo las filas con 'Año' menor a 2010
```

df\_filtrado



	Año	Defunciones por cáncer de mama en mujeres de 25 años o más	Población de mujeres de 25 años o más	Tasa de mortalidad por cáncer de mama
0	1990	4380	35852664	12.216665
1	1991	4668	37007840	12.613544
2	1992	5036	38193064	13.185640
3	1993	5352	39392350	13.586394
4	1994	5478	40623620	13.484766
5	1995	5960	41890024	14.227731
6	1996	6130	43182190	14.195667
7	1997	6362	44490494	14.299684
8	1998	6692	45816222	14.606180
9	1999	6764	47132266	14.351103
10	2000	6838	48432714	14.118556
11	2001	7126	49756872	14.321640
12	2002	7644	51131594	14.949661
13	2003	7722	52529180	14.700401
14	2004	8300	53906692	15.396975
15	2005	8410	55250834	15.221490
16	2006	8880	56601780	15.688553
17	2007	9162	57999644	15.796649
18	2008	9604	59435352	16.158733

```
# Concatenar las filas de df_grouped2 a df_filtrado
df_combinado = pd.concat([df_filtrado, df_grouped2], ignore_index=True)
```

```
# Ver el DataFrame resultante
df_combinado
```



Año	Defunciones por cáncer de mama en mujeres de 25 años o más	Población de mujeres de 25 años o más	Tasa de mortalidad por cáncer de mama
0	1990	4380	12.216665
1	1991	4668	12.613544
2	1992	5036	13.185640
3	1993	5352	13.586394
4	1994	5478	13.484766
5	1995	5960	14.227731
6	1996	6130	14.195667
7	1997	6362	14.299684
8	1998	6692	14.606180
9	1999	6764	14.351103
10	2000	6838	14.118556
11	2001	7126	14.321640
12	2002	7644	14.949661
13	2003	7722	14.700401
14	2004	8300	15.396975
15	2005	8410	15.221490
16	2006	8880	15.688553
17	2007	9162	15.796649
18	2008	9604	16.158733
19	2009	9786	16.073646
20	2010	10074	16.168903
21	2011	10414	16.345929
22	2012	11168	17.150725
23	2013	11052	16.614223
24	2014	11948	17.587295
25	2015	12552	18.097296

```
#montar drive
from google.colab import drive
drive.mount('/content/drive')
```

→ Mounted at /content/drive

```
# Especifica la ruta donde deseas guardar el archivo (ajusta la carpeta y nombre)
ruta_guardado = '/content/drive/MyDrive/Cancer de mama/Tasa de mortalidad de 1990 - 2013 en'

# Guardar el DataFrame como archivo CSV
df_combinado.to_csv(ruta_guardado, index=False)

print("Archivo guardado en: ", ruta_guardado)
```

→ -----

```
NameError Traceback (most recent call last)
<ipython-input-6-c0cbc0f8ee34> in <cell line: 5>()
      3
      4 # Guardar el DataFrame como archivo CSV
----> 5 df_combinado.to_csv(ruta_guardado, index=False)
      6
      7 print("Archivo guardado en: ", ruta_guardado)

NameError: name 'df_combinado' is not defined
```

```
import pandas as pd
ruta_guardado = '/content/drive/MyDrive/Cancer de mama/Tasa de mortalidad de 1990 - 2013 en'
df = pd.read_csv(ruta_guardado)
df
```



Año	Defunciones por cáncer de mama en mujeres de 25 años o más	Población de mujeres de 25 años o más	Tasa de mortalidad por cáncer de mama
0 1990	4380	35852664	12
1 1991	4668	37007840	13
2 1992	5036	38193064	13
3 1993	5352	39392350	14
4 1994	5478	40623620	13
5 1995	5960	41890024	14
6 1996	6130	43182190	14
7 1997	6362	44490494	14
8 1998	6692	45816222	15
9 1999	6764	47132266	14
10 2000	6838	48432714	14
11 2001	7126	49756872	14
12 2002	7644	51131594	15
13 2003	7722	52529180	15
14 2004	8300	53906692	15
15 2005	8410	55250834	15
16 2006	8880	56601780	16
17 2007	9162	57999644	16
18 2008	9604	59435352	16
19 2009	9786	60882268	16
20 2010	10074	62304782	16
21 2011	10414	63710053	16
22 2012	11168	65116779	17
23 2013	11052	66521317	17
24 2014	11948	67935405	18
25 2015	12552	69358429	18

```
import matplotlib.pyplot as plt
```

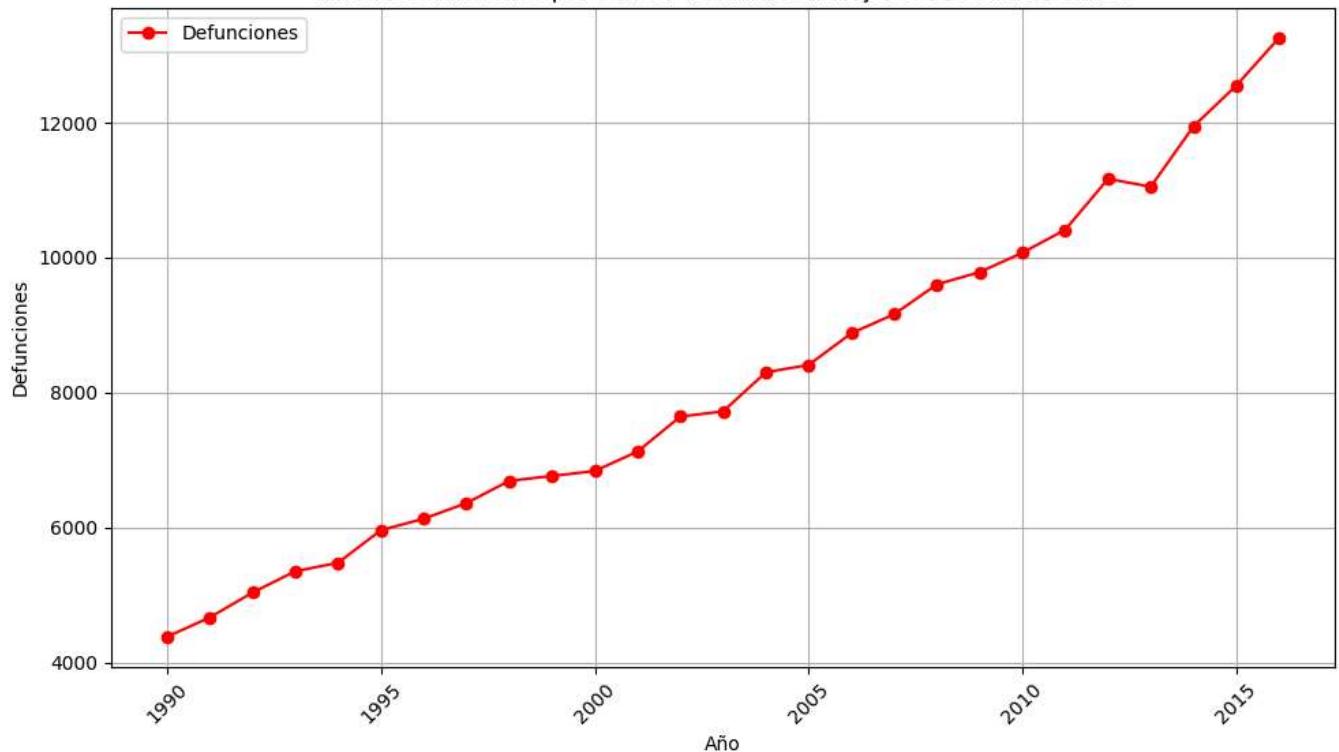
```
# Graficar Año vs Defunciones por Cáncer de Mama en Mujeres de 25 Años o Más
plt.figure(figsize=(10,6))
plt.plot(df['Año'], df['Defunciones por cáncer de mama en mujeres de 25 años o más'], color='red')
plt.title('Año vs Defunciones por Cáncer de Mama en Mujeres de 25 Años o Más')
plt.xlabel('Año')
plt.ylabel('Defunciones')
plt.grid(True)
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

# Graficar Año vs Población de Mujeres de 25 Años o Más
plt.figure(figsize=(10,6))
plt.plot(df['Año'], df['Población de mujeres de 25 años o más'], color='blue', marker='o', linestyle='dashed')
plt.title('Año vs Población de Mujeres de 25 Años o Más')
plt.xlabel('Año')
plt.ylabel('Población')
plt.grid(True)
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

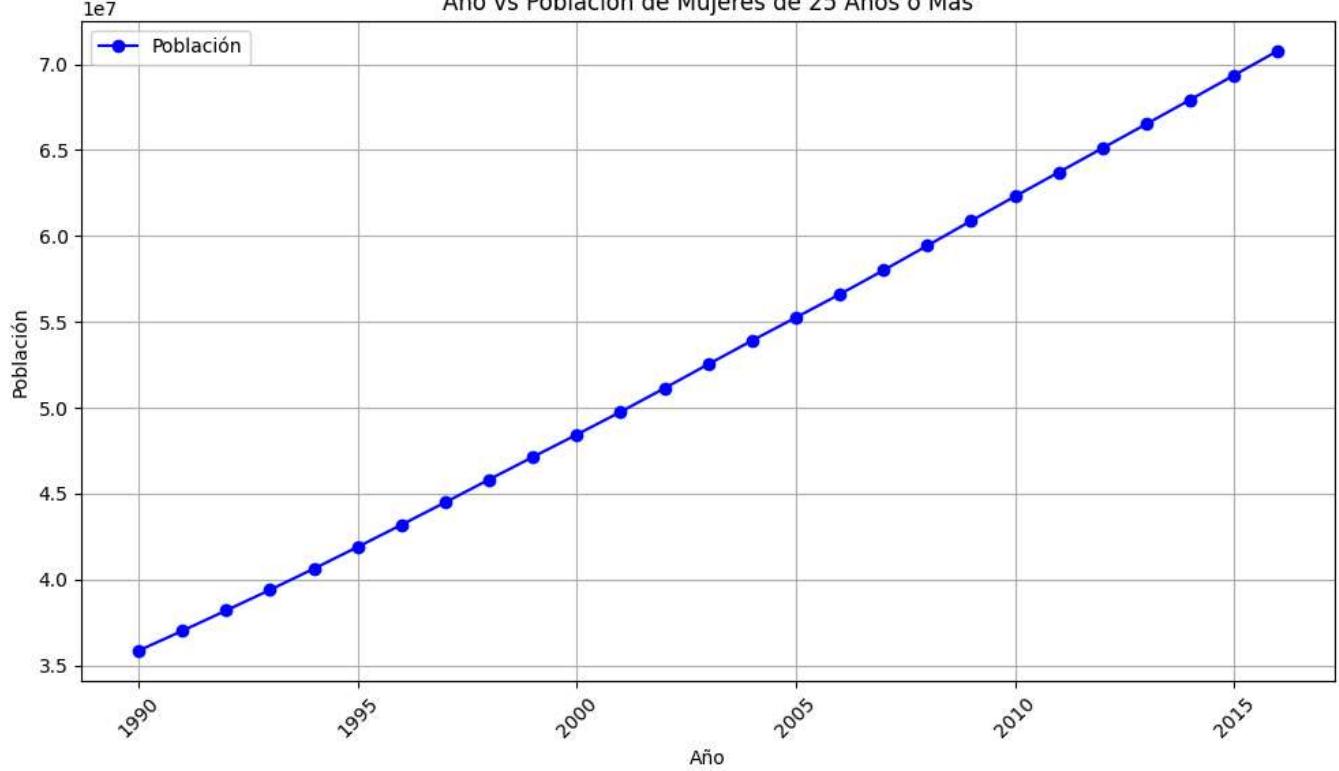
# Graficar Año vs Tasa de Mortalidad por Cáncer de Mama
plt.figure(figsize=(10,6))
plt.plot(df['Año'], df['Tasa de mortalidad por cáncer de mama'], color='green', marker='o', linestyle='dashed')
plt.title('Año vs Tasa de Mortalidad por Cáncer de Mama')
plt.xlabel('Año')
plt.ylabel('Tasa de Mortalidad')
plt.grid(True)
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()
```



## Año vs Defunciones por Cáncer de Mama en Mujeres de 25 Años o Más

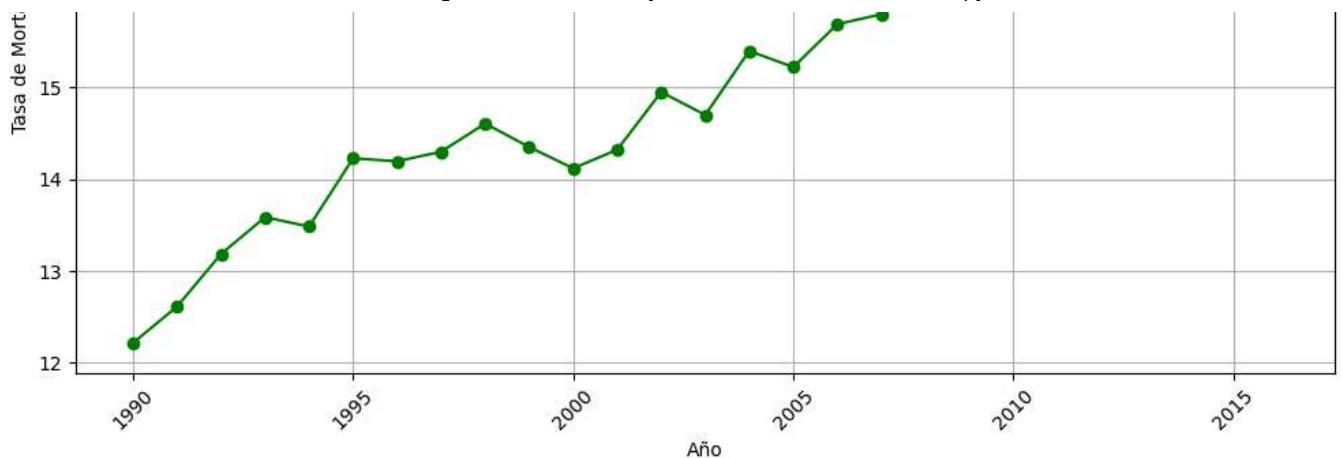


## Año vs Población de Mujeres de 25 Años o Más



## Año vs Tasa de Mortalidad por Cáncer de Mama





```
import matplotlib.pyplot as plt

# Filtrar el DataFrame para los años 2013 a 2016
df_filtered = df[(df['Año'] >= 2013) & (df['Año'] <= 2016)]

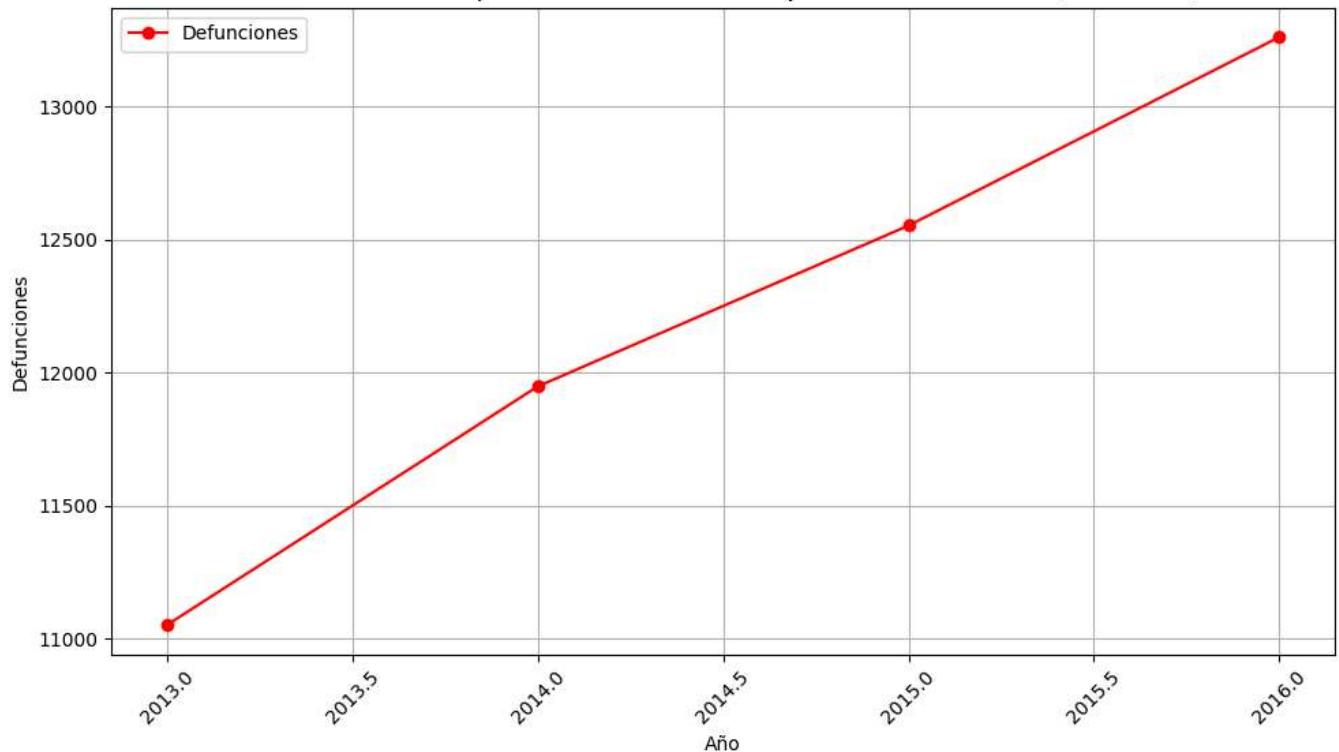
# Graficar Año vs Defunciones por Cáncer de Mama en Mujeres de 25 Años o Más
plt.figure(figsize=(10,6))
plt.plot(df_filtered['Año'], df_filtered['Defunciones por cáncer de mama en mujeres de 25 aÑos o mÁs'])
plt.title('Año vs Defunciones por Cáncer de Mama en Mujeres de 25 Años o Más (2013-2016)')
plt.xlabel('Año')
plt.ylabel('Defunciones')
plt.grid(True)
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

# Graficar Año vs Población de Mujeres de 25 Años o Más
plt.figure(figsize=(10,6))
plt.plot(df_filtered['Año'], df_filtered['Población de mujeres de 25 años o más'], color='blue')
plt.title('Año vs Población de Mujeres de 25 Años o Más (2013-2016)')
plt.xlabel('Año')
plt.ylabel('Población')
plt.grid(True)
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()

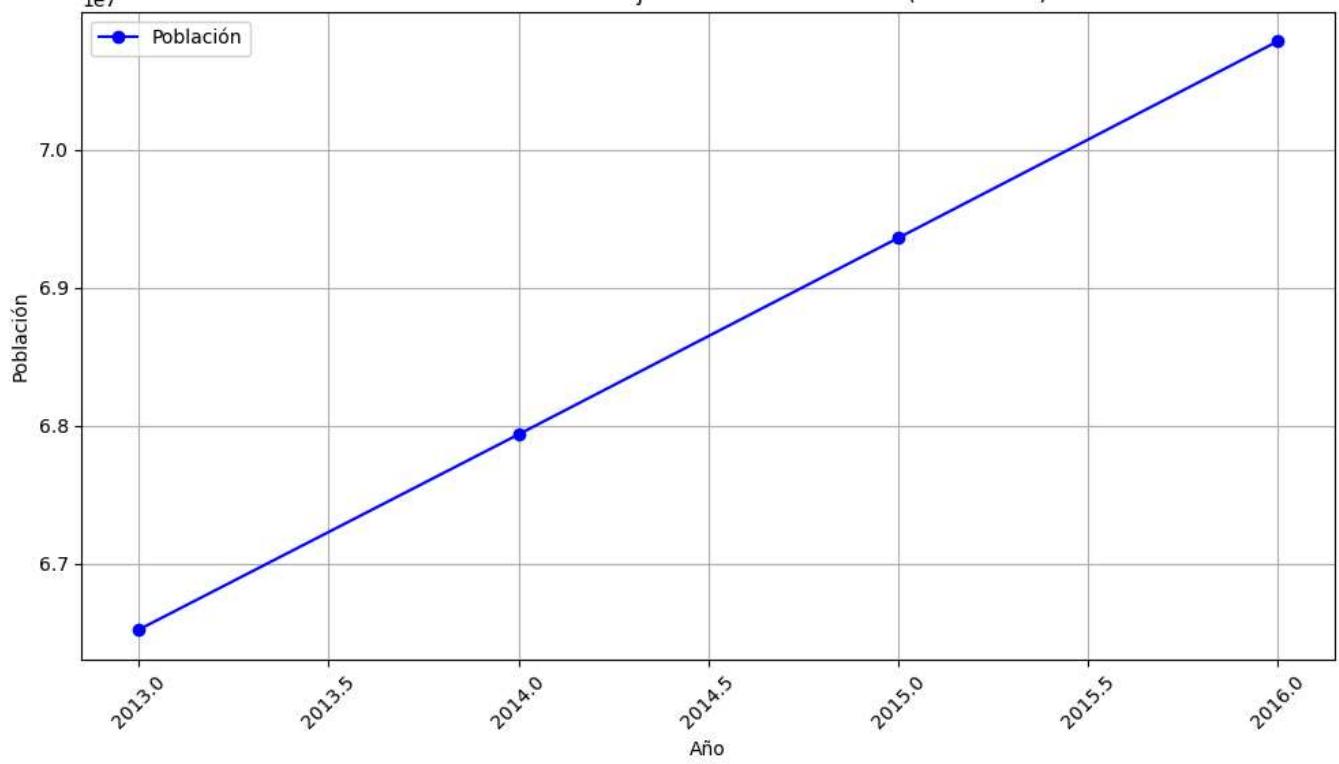
# Graficar Año vs Tasa de Mortalidad por Cáncer de Mama
plt.figure(figsize=(10,6))
plt.plot(df_filtered['Año'], df_filtered['Tasa de mortalidad por cáncer de mama'], color='green')
plt.title('Año vs Tasa de Mortalidad por Cáncer de Mama (2013-2016)')
plt.xlabel('Año')
plt.ylabel('Tasa de Mortalidad')
plt.grid(True)
plt.xticks(rotation=45)
plt.legend()
plt.tight_layout()
plt.show()
```



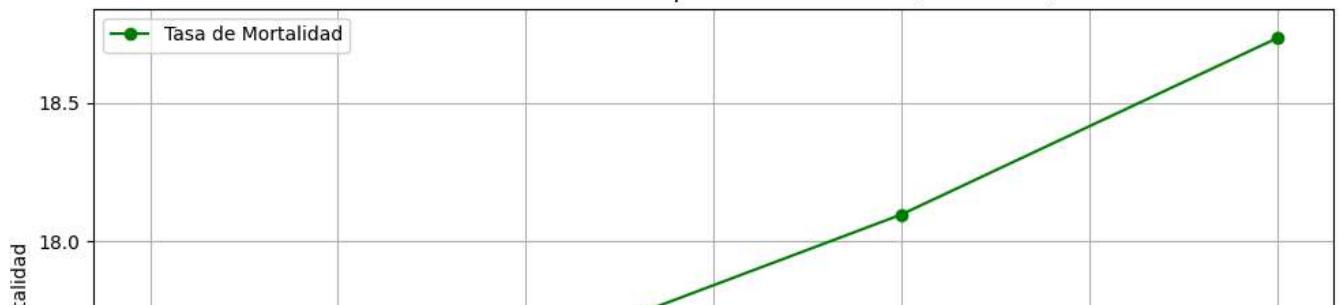
## Año vs Defunciones por Cáncer de Mama en Mujeres de 25 Años o Más (2013-2016)

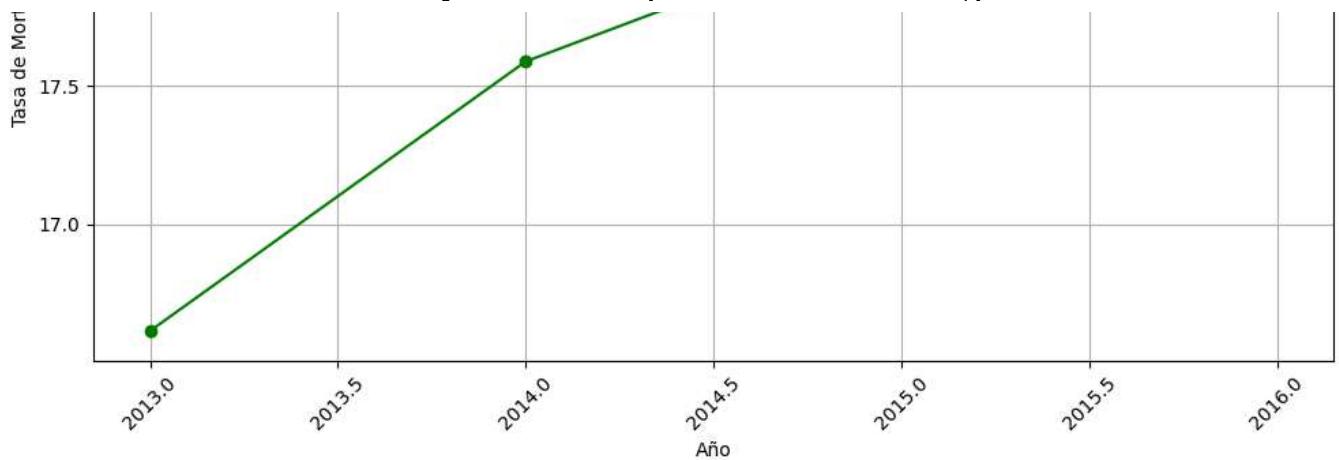


## Año vs Población de Mujeres de 25 Años o Más (2013-2016)



## Año vs Tasa de Mortalidad por Cáncer de Mama (2013-2016)





```

import pandas as pd

# Crear el DataFrame con los datos proporcionados
data = {
    "Año": [2010, 2011, 2012, 2013, 2014, 2015, 2016],
    "Defunciones por cáncer de mama en mujeres de 25 años o más": [5024, 5206, 5583, 5526, 5974, 6252, 6629],
    "Población de mujeres de 25 años o más": [
        30844960, 31604691, 32362693, 33119331, 33873575, 34611681, 35344978
    ],
    "Tasa de mortalidad por cáncer de mama": [
        16.3203324, 16.47223825, 17.25134555, 16.68511964, 17.63616624, 18.06326598, 18.7551
    ],
}
df2 = pd.DataFrame(data)
df2

```

	Año	Defunciones por cáncer de mama en mujeres de 25 años o más	Población de mujeres de 25 años o más	Tasa de mortalidad por cáncer de mama
0	2010	5024	30844960	16.320332
1	2011	5206	31604691	16.472238
2	2012	5583	32362693	17.251346
3	2013	5526	33119331	16.685120
4	2014	5974	33873575	17.636166
5	2015	6252	34611681	18.063266
6	2016	6629	35344978	18.755140

```

import matplotlib.pyplot as plt

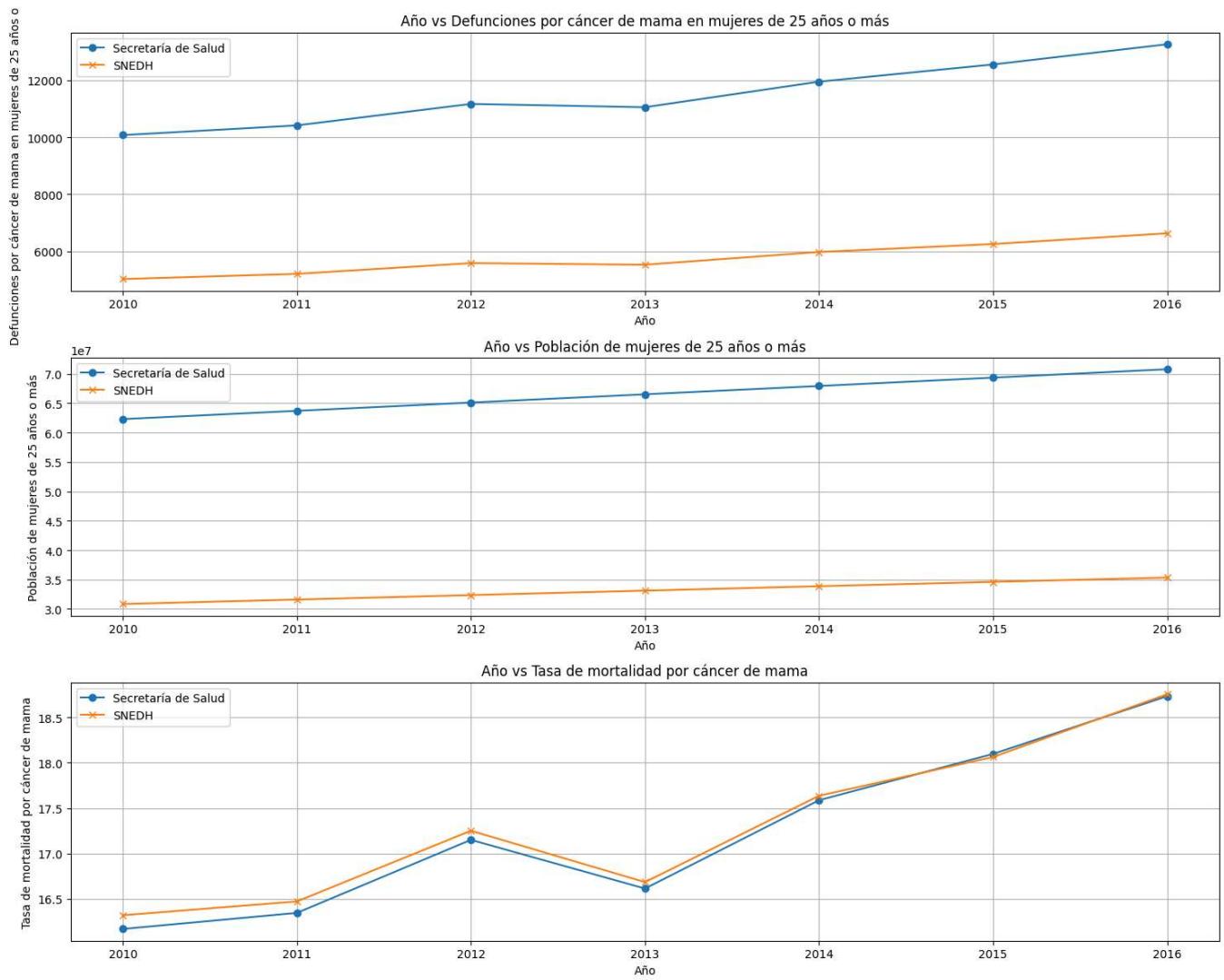
# Filtrar el DataFrame df para que solo contenga datos entre 2010 y 2016
df_filtered = df[(df["Año"] >= 2010) & (df["Año"] <= 2016)]

# Crear las gráficas comparativas
columns_to_compare = [
    "Defunciones por cáncer de mama en mujeres de 25 años o más",
    "Población de mujeres de 25 años o más",
    "Tasa de mortalidad por cáncer de mama",
]

plt.figure(figsize=(15, 12))
for i, column in enumerate(columns_to_compare, 1):
    plt.subplot(3, 1, i)
    plt.plot(df_filtered["Año"], df_filtered[column], label="Secretaría de Salud", marker="c")
    plt.plot(df2["Año"], df2[column], label="SNEDH", marker="x")

```

```
plt.title(f"Año vs {column}")  
plt.xlabel("Año")  
plt.ylabel(column)  
plt.legend()  
plt.grid()  
  
plt.tight_layout()  
plt.show()
```



```
# Comparar tasa de mortalidad
df_comparison = df_filtered.merge(df2, on="Año", suffixes=("_salud", "_snedh"))

# Calcular la precisión para la tasa de mortalidad
df_comparison["Diferencia Tasa de Mortalidad (%)" ] = abs(
    df_comparison["Tasa de mortalidad por cáncer de mama_salud"]
    - df_comparison["Tasa de mortalidad por cáncer de mama_snedh"])
) / df_comparison["Tasa de mortalidad por cáncer de mama_salud"] * 100

precision_tasa = 100 - df_comparison["Diferencia Tasa de Mortalidad (%)" ].mean()

# Calcular las diferencias relativas para población y defunciones
df_comparison["Diferencia Población (%)" ] = abs(
    df_comparison["Población de mujeres de 25 años o más_salud"]
    - df_comparison["Población de mujeres de 25 años o más_snedh"])
) / df_comparison["Población de mujeres de 25 años o más_salud"] * 100

df_comparison["Diferencia Defunciones (%)" ] = abs(
    df_comparison["Defunciones por cáncer de mama en mujeres de 25 años o más_salud"]
    - df_comparison["Defunciones por cáncer de mama en mujeres de 25 años o más_snedh"])
) / df_comparison["Defunciones por cáncer de mama en mujeres de 25 años o más_salud"] * 100

# Promedio de diferencias relativas
diferencia_poblacion = df_comparison["Diferencia Población (%)" ].mean()
diferencia_defunciones = df_comparison["Diferencia Defunciones (%)" ].mean()

# Resultados
precision_tasa, diferencia_poblacion, diferencia_defunciones
```

→ (99.52769300201372, 50.24307315854816, 50.04947824950068)

```
import pandas as pd

# Data for the population of women over 25 years from 1990 to 2020
data = {
    "Año": [1990, 1995, 2000, 2005, 2010, 2020],
    "Población de mujeres mayores de 25 años": [16690364, 20275132, 22948793, 26195454, 3018]
}

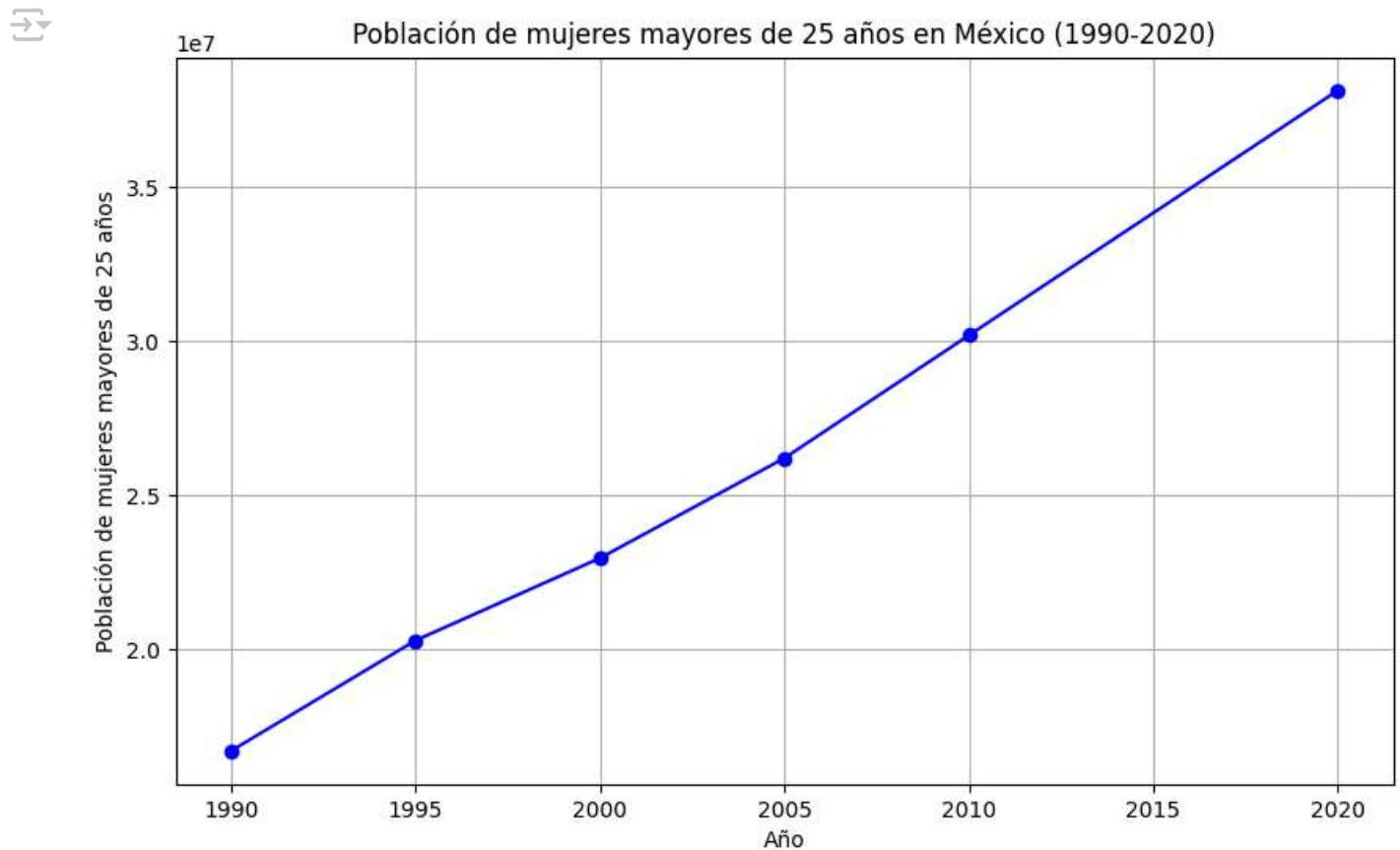
# Create the DataFrame
df_poblacion_mujeres = pd.DataFrame(data)
```

```
# Display the DataFrame  
print(df_poblacion_mujeres)
```

→ Año Población de mujeres mayores de 25 años

	Año	Población de mujeres mayores de 25 años
0	1990	16690364
1	1995	20275132
2	2000	22948793
3	2005	26195454
4	2010	30181997
5	2020	38132240

```
# Crear el gráfico  
plt.figure(figsize=(10, 6))  
plt.plot(df_poblacion_mujeres["Año"], df_poblacion_mujeres["Población de mujeres mayores de 25 años"])  
plt.title("Población de mujeres mayores de 25 años en México (1990-2020)")  
plt.xlabel("Año")  
plt.ylabel("Población de mujeres mayores de 25 años")  
plt.grid(True)  
plt.show()
```



```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

# Datos existentes de la población de mujeres mayores de 25 años
data = {
    "Año": [1990, 1995, 2000, 2005, 2010, 2020],
    "Población de mujeres mayores de 25 años": [16690364, 20275132, 22948793, 26195454, 3018
}

df_poblacion_mujeres = pd.DataFrame(data)

# Crear un rango de años de 1990 a 2023
years = np.arange(1990, 2024).reshape(-1, 1)

# Extraer los valores de los años disponibles y las poblaciones disponibles
available_years = df_poblacion_mujeres["Año"].values.reshape(-1, 1)
available_population = df_poblacion_mujeres["Población de mujeres mayores de 25 años"].values

# Crear y entrenar el modelo de regresión lineal
model = LinearRegression()
model.fit(available_years, available_population)

# Predecir la población para los años de 1990 a 2023
predicted_population = model.predict(years)

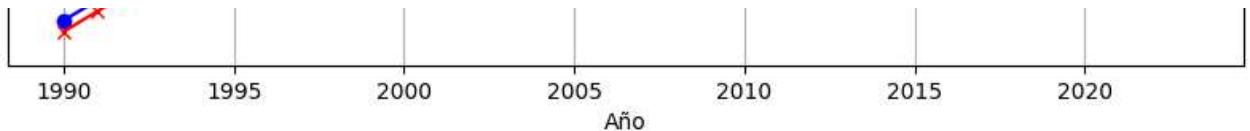
# Crear un DataFrame con los años y las poblaciones predichas
df_predicted = pd.DataFrame({
    "Año": years.flatten(),
    "Población de mujeres mayores de 25 años (predicha)": predicted_population
})

# Mostrar el DataFrame con las predicciones
print(df_predicted)

# Graficar los resultados
plt.figure(figsize=(10, 6))
plt.plot(df_poblacion_mujeres["Año"], df_poblacion_mujeres["Población de mujeres mayores de 25 años"])
plt.plot(df_predicted["Año"], df_predicted["Población de mujeres mayores de 25 años (predicha)"])
plt.title("Población de mujeres mayores de 25 años (1990-2023) - Predicción")
plt.xlabel("Año")
plt.ylabel("Población de mujeres mayores de 25 años")
plt.legend()
plt.grid(True)
plt.show()
```

	Año	Población de mujeres mayores de 25 años (predicha)
0	1990	1.631068e+07
1	1991	1.701767e+07
2	1992	1.772467e+07
3	1993	1.843167e+07
4	1994	1.913867e+07
5	1995	1.984567e+07
6	1996	2.055267e+07
7	1997	2.125967e+07
8	1998	2.196667e+07
9	1999	2.267367e+07
10	2000	2.338067e+07
11	2001	2.408767e+07
12	2002	2.479466e+07
13	2003	2.550166e+07
14	2004	2.620866e+07
15	2005	2.691566e+07
16	2006	2.762266e+07
17	2007	2.832966e+07
18	2008	2.903666e+07
19	2009	2.974366e+07
20	2010	3.045066e+07
21	2011	3.115766e+07
22	2012	3.186466e+07
23	2013	3.257165e+07
24	2014	3.327865e+07
25	2015	3.398565e+07
26	2016	3.469265e+07
27	2017	3.539965e+07
28	2018	3.610665e+07
29	2019	3.681365e+07
30	2020	3.752065e+07
31	2021	3.822765e+07
32	2022	3.893465e+07
33	2023	3.964165e+07





```
import xgboost as xgb
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Datos existentes de la población de mujeres mayores de 25 años
data = {
    "Año": [1990, 1995, 2000, 2005, 2010, 2020],
    "Población de mujeres mayores de 25 años": [16690364, 20275132, 22948793, 26195454, 3018
}
df_poblacion_mujeres = pd.DataFrame(data)

# Crear un rango de años de 1990 a 2023
years = np.arange(1990, 2024).reshape(-1, 1)

# Extraer los valores de los años disponibles y las poblaciones disponibles
available_years = df_poblacion_mujeres["Año"].values.reshape(-1, 1)
available_population = df_poblacion_mujeres["Población de mujeres mayores de 25 años"].values

# Convertir los datos a formato DMatrix (requerido por XGBoost)
dtrain = xgb.DMatrix(available_years, label=available_population)

# Establecer parámetros para el modelo de XGBoost
params = {
    "objective": "reg:squarederror", # Regresión
    "colsample_bytree": 0.3,
    "learning_rate": 0.1,
    "max_depth": 5,
    "alpha": 10,
    "n_estimators": 100
}

# Entrenar el modelo
xgb_model = xgb.train(params, dtrain, num_boost_round=100)

# Predecir la población para los años de 1990 a 2023
dtest = xgb.DMatrix(years)
predicted_population_xgb = xgb_model.predict(dtest)
```

```
# Crear un DataFrame con los años y las poblaciones predichas
df_predicted_xgb = pd.DataFrame({
    "Año": years.flatten(),
    "Población de mujeres mayores de 25 años (predicha)": predicted_population_xgb
})

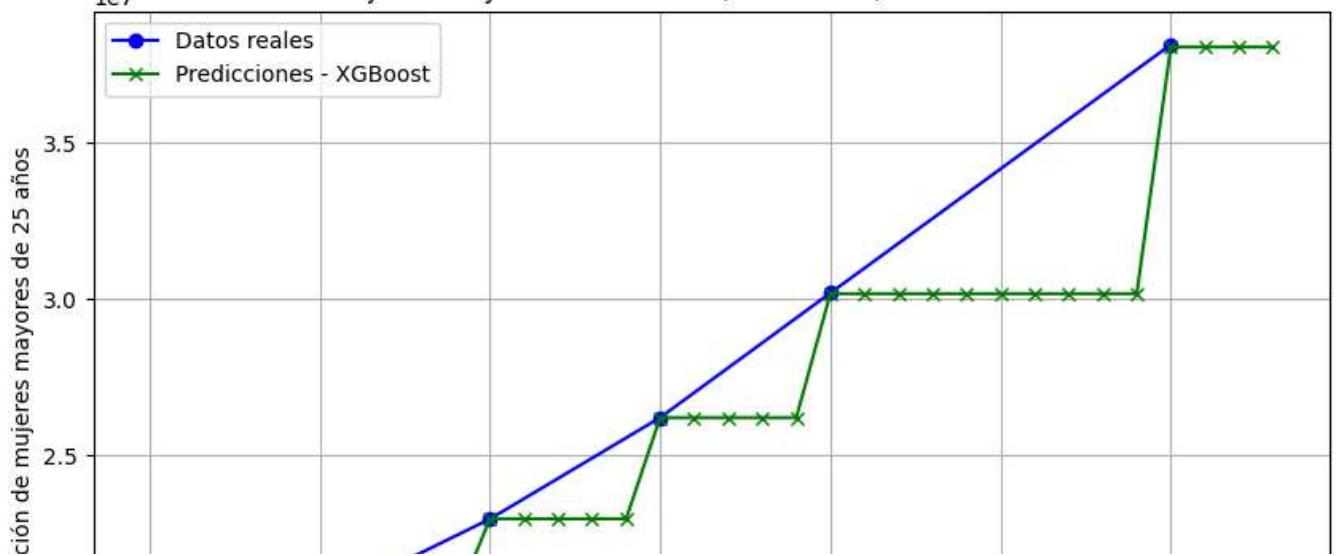
# Mostrar el DataFrame con las predicciones
print(df_predicted_xgb)

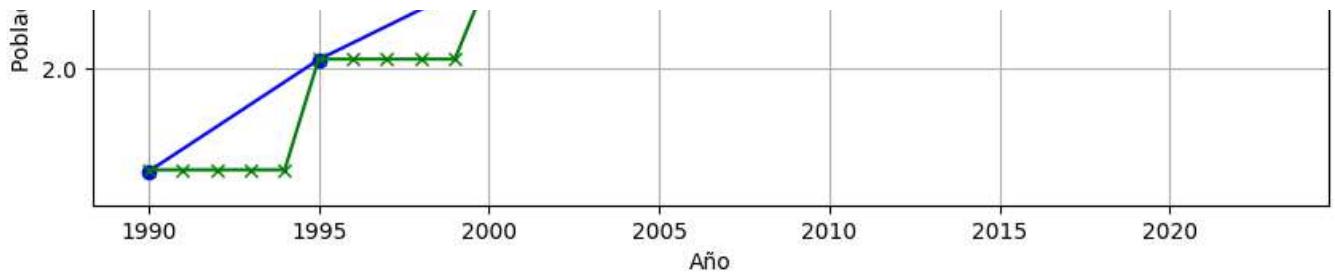
# Graficar los resultados
plt.figure(figsize=(10, 6))
plt.plot(df_poblacion_mujeres["Año"], df_poblacion_mujeres["Población de mujeres mayores de 25 años"])
plt.plot(df_predicted_xgb["Año"], df_predicted_xgb["Población de mujeres mayores de 25 años"])
plt.title("Población de mujeres mayores de 25 años (1990-2023) - Predicción usando XGBoost")
plt.xlabel("Año")
plt.ylabel("Población de mujeres mayores de 25 años")
plt.legend()
plt.grid(True)
plt.show()
```

/usr/local/lib/python3.10/dist-packages/xgboost/core.py:158: UserWarning: [04:47:18] WAF  
Parameters: { "n\_estimators" } are not used.

```
warnings.warn(smsg, UserWarning)
    Año   Población de mujeres mayores de 25 años (predicha)
0    1990                      16745287.0
1    1991                      16745287.0
2    1992                      16745287.0
3    1993                      16745287.0
4    1994                      16745287.0
5    1995                     20288974.0
6    1996                     20288974.0
7    1997                     20288974.0
8    1998                     20288974.0
9    1999                     20288974.0
10   2000                     22954018.0
11   2001                     22954018.0
12   2002                     22954018.0
13   2003                     22954018.0
14   2004                     22954018.0
15   2005                     26192728.0
16   2006                     26192728.0
17   2007                     26192728.0
18   2008                     26192728.0
19   2009                     26192728.0
20   2010                     30162162.0
21   2011                     30162162.0
22   2012                     30162162.0
23   2013                     30162162.0
24   2014                     30162162.0
25   2015                     30162162.0
26   2016                     30162162.0
27   2017                     30162162.0
28   2018                     30162162.0
29   2019                     30162162.0
30   2020                     38057508.0
31   2021                     38057508.0
32   2022                     38057508.0
33   2023                     38057508.0
```

1e7 Población de mujeres mayores de 25 años (1990-2023) - Predicción usando XGBoost





```

import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

# Datos de población
data = {
    "Año": [1895, 1900, 1910, 1920, 1940, 1950, 1960, 1970, 1980, 1990, 1995, 2000, 2005, 2010, 2015, 2020],
    "Población de mujeres mayores de 25 años": [
        2456175, 3314941, 3673909, 3754899, 4161317, 5162134, 6544979, 8564920, 12354763,
        16690364, 20275132, 22948793, 26195454, 30181997, 38132240
    ]
}

# Crear DataFrame
df_poblacion_mujeres = pd.DataFrame(data)

# Crear el gráfico de dispersión
plt.figure(figsize=(10, 6))
plt.scatter(df_poblacion_mujeres['Año'], df_poblacion_mujeres['Población de mujeres mayores de 25 años'])

# Añadir formato de etiquetas sin notación científica
plt.gca().yaxis.set_major_formatter(FuncFormatter(lambda x, _: '{:.0f}'.format(x)))

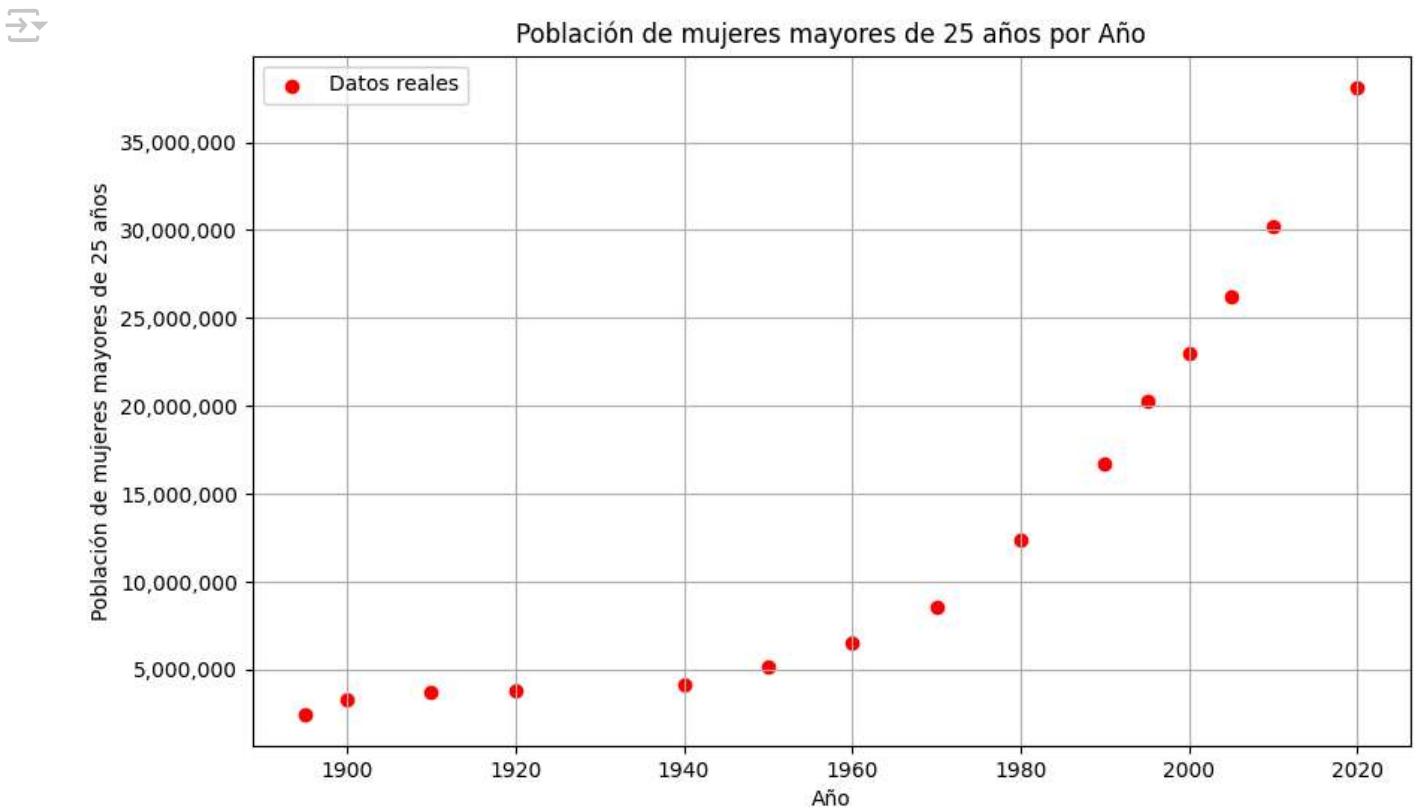
# Añadir etiquetas y título
plt.xlabel('Año')
plt.ylabel('Población de mujeres mayores de 25 años')
plt.title('Población de mujeres mayores de 25 años por Año')

# Mostrar la leyenda
plt.legend()

# Añadir una cuadrícula
plt.grid(True)

```

```
# Mostrar la gráfica
plt.show()
```



```
import pandas as pd
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

# Configurar pandas y numpy para evitar notación científica
pd.set_option('display.float_format', '{:.0f}'.format)
np.set_printoptions(suppress=True)

# Datos de población
data = {
    "Año": [1895, 1900, 1910, 1920, 1940, 1950, 1960, 1970, 1980, 1990, 1995, 2000, 2005, 2010, 2015, 2020],
    "Población de mujeres mayores de 25 años": [
        2456175, 3314941, 3673909, 3754899, 4161317, 5162134, 6544979, 8564920, 12354763,
        16690364, 20275132, 22948793, 26195454, 30181997, 38132240
    ]
}
```

```
]
}

# Crear DataFrame
df = pd.DataFrame(data)

# Preparar los datos
X = df[["Año"]]
y = df["Población de mujeres mayores de 25 años"]

# Crear características polinómicas (grado 3, por ejemplo)
poly = PolynomialFeatures(degree=3, include_bias=False)
X_poly = poly.fit_transform(X)

# Ajustar el modelo de regresión
model = LinearRegression()
model.fit(X_poly, y)

# Predicción para todo el rango de años
years = np.arange(1895, 2023 + 1).reshape(-1, 1)
years_poly = poly.transform(years)
predictions = model.predict(years_poly)

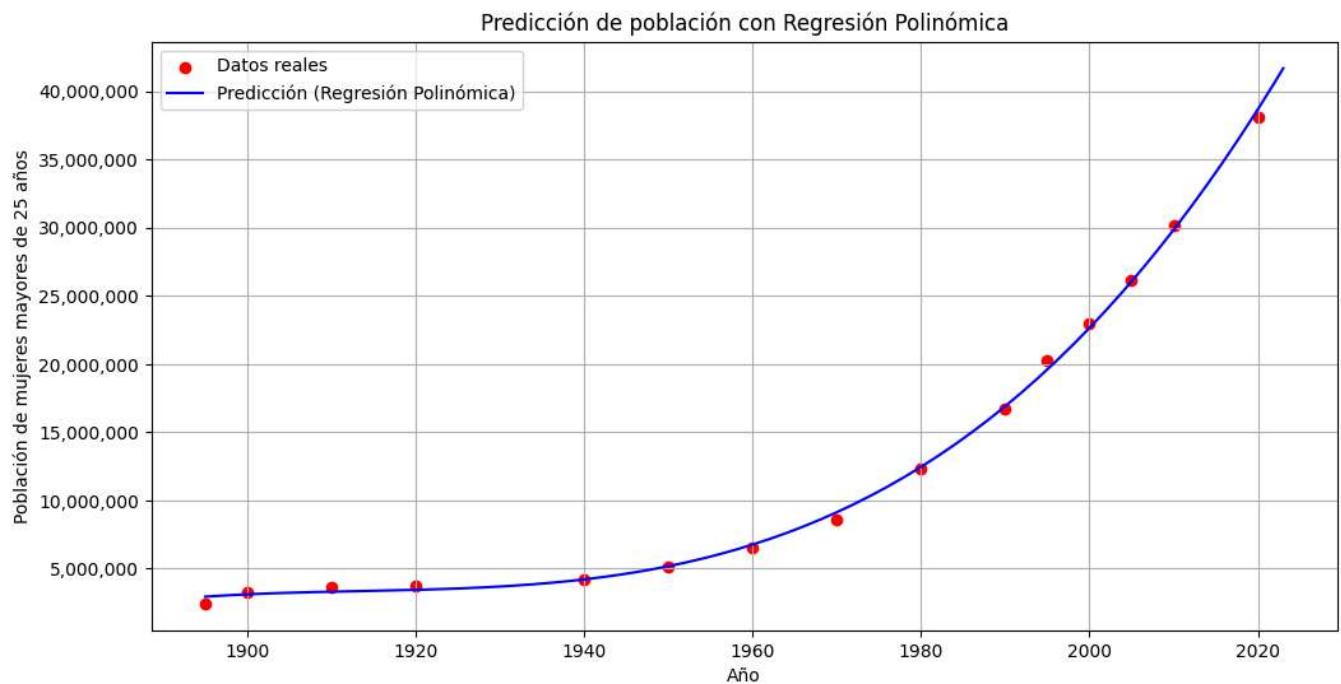
# Graficar los resultados
plt.figure(figsize=(12, 6))
plt.scatter(X, y, color="red", label="Datos reales")
plt.plot(years, predictions, color="blue", label="Predicción (Regresión Polinómica)")
plt.xlabel("Año")
plt.ylabel("Población de mujeres mayores de 25 años")
plt.title("Predicción de población con Regresión Polinómica")
plt.legend()
plt.grid()

# Formatear el eje y para evitar notación científica
formatter = FuncFormatter(lambda x, _: f'{x:.0f}')
plt.gca().yaxis.set_major_formatter(formatter)

plt.show()

# Resultados predichos
df_predictions = pd.DataFrame({"Año": years.flatten(), "Población estimada": predictions})
print(df_predictions) # Mostrar primeras filas
```

```
→ /usr/local/lib/python3.10/dist-packages/sklearn/base.py:493: UserWarning: X does not have
warnings.warn(
```



[129 rows x 2 columns]

df\_predictions

	Año	Población estimada
0	1895	2948882
1	1896	2988943
2	1897	3026038
3	1898	3060337
4	1899	3092009
...	...	...
124	2019	37732257
125	2020	38693873
126	2021	39673487
127	2022	40671268
128	2023	41687386

129 rows × 2 columns

```
import pandas as pd

# Ruta al archivo CSV en Google Drive
file_path = '/content/drive/MyDrive/Cancer de mama/SNEDH_SdR05a_FichaEvidencias.csv'

# Cargar el CSV en un DataFrame
df_SNEDH = pd.read_csv(file_path,skiprows=9)

# Ver las primeras filas del DataFrame
df_SNEDH
```



	Cobertura geografica	Periodo	Tmcama	Defcama	Pfem
0	Estados Unidos Mexicanos	2010	16	5034	30844960
1	Aguascalientes	2010	14	45	310718
2	Baja California	2010	20	163	825760
3	Baja California Sur	2010	19	31	163749
4	Campeche	2010	12	26	218399
...	...	...	...	...	...
391	Tamaulipas	2021	28	300	1081715
392	Tlaxcala	2021	12	48	404458
393	Veracruz de Ignacio de la Llave	2021	20	532	2603056
394	Yucatán	2021	15	107	720832
395	Zacatecas	2021	21	97	468697

396 rows × 5 columns

```
# Contar cuántas veces aparece 'Estados Unidos Mexicanos' en la columna 'Cobertura geografic
conteo_estados_unidos_mexicanos = df_SNEDH[df_SNEDH['Cobertura geografica'] == 'Estados Unic

# Mostrar el resultado
print(conteo_estados_unidos_mexicanos)
```

 12

```
# Filtrar las filas donde la columna 'Cobertura geografica' es 'Estados Unidos Mexicanos'
filas_estados_unidos_mexicanos = df_SNEDH[df_SNEDH['Cobertura geografica'] == 'Estados Unidc

# Eliminar las columnas 'Cobertura geografica', 'Tmcama' y 'Defcama'
filas_estados_unidos_mexicanos = filas_estados_unidos_mexicanos.drop(columns=['Cobertura gec

# Renombrar la columna 'Periodo' a 'Año'
filas_estados_unidos_mexicanos = filas_estados_unidos_mexicanos.rename(columns={'Periodo': 'A

# Reindexar para restablecer el índice
filas_estados_unidos_mexicanos = filas_estados_unidos_mexicanos.reset_index(drop=True)

# Mostrar las filas filtradas y modificadas
filas_estados_unidos_mexicanos
```

	Año	Pfem
0	2010	30844960
1	2011	31604691
2	2012	32362693
3	2013	33119331
4	2014	33873575
5	2015	34611681
6	2016	35344978
7	2017	36075493
8	2018	36797671
9	2019	37511160
10	2020	38325557
11	2021	38695271

```
import pandas as pd
ruta_guardado = '/content/drive/MyDrive/Cancer de mama/Tasa de mortalidad de 1990 - 2013 en
df = pd.read_csv(ruta_guardado)
df
```



Año	Defunciones por cáncer de mama en mujeres de 25 años o más	Población de mujeres de 25 años o más	Tasa de mortalidad por cáncer de mama
0	1990	4380	12.216665
1	1991	4668	12.613544
2	1992	5036	13.185640
3	1993	5352	13.586394
4	1994	5478	13.484766
5	1995	5960	14.227731
6	1996	6130	14.195667
7	1997	6362	14.299684
8	1998	6692	14.606180
9	1999	6764	14.351103
10	2000	6838	14.118556
11	2001	7126	14.321640
12	2002	7644	14.949661
13	2003	7722	14.700401
14	2004	8300	15.396975
15	2005	8410	15.221490
16	2006	8880	15.688553
17	2007	9162	15.796649
18	2008	9604	16.158733
19	2009	9786	16.073646
20	2010	10074	16.168903
21	2011	10414	16.345929
22	2012	11168	17.150725
23	2013	11052	16.614223
24	2014	11948	17.587295
25	2015	12552	18.097296

df\_predictions

	Año	Población estimada
0	1895	2948882
1	1896	2988943
2	1897	3026038
3	1898	3060337
4	1899	3092009
...	...	...
124	2019	37732257
125	2020	38693873
126	2021	39673487
127	2022	40671268
128	2023	41687386

129 rows × 2 columns

```

import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

# Asegúrate de que todos los DataFrames tienen la columna 'Año' y que los valores se alinear
# Para esto, primero nos aseguramos de que los DataFrames estén ordenados por 'Año'
df_SNEDH_sorted = filas_estados_unidos_mexicanos.sort_values(by="Año")
df_sorted = df.sort_values(by="Año")
df_predictions_sorted = df_predictions.sort_values(by="Año")

# Graficar los datos
plt.figure(figsize=(12, 6))

# Graficamos los datos para cada DataFrame
plt.plot(df_SNEDH_sorted["Año"], df_SNEDH_sorted["Pfem"], color="red", label="Datos SNEDH")
plt.plot(df_sorted["Año"], df_sorted["Población de mujeres de 25 años o más"], color="blue", label="Población de mujeres de 25 años o más")
plt.plot(df_predictions_sorted["Año"], df_predictions_sorted["Población estimada"], color="green", label="Población estimada")

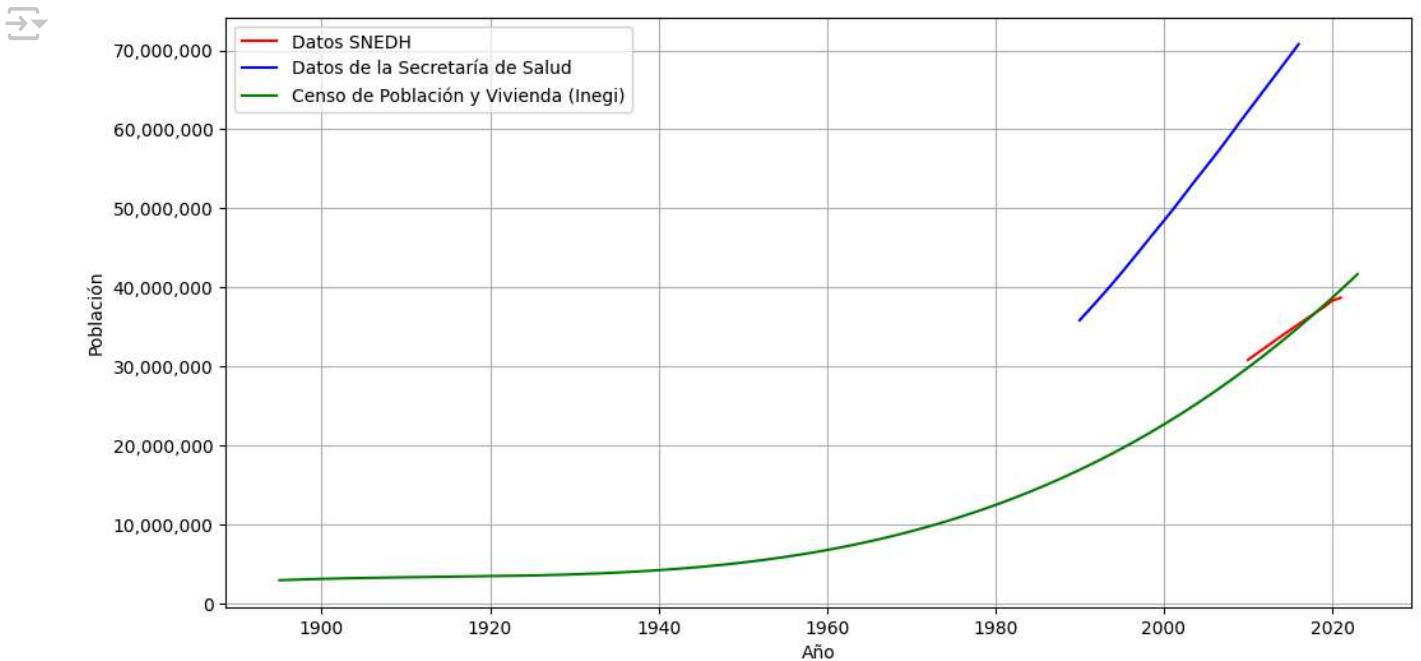
# Añadir etiquetas y título
plt.xlabel("Año")
plt.ylabel("Población")
plt.title("Comparación de set de datos de la Población Femenina de 25 años o más en México")

# Añadir leyenda
plt.legend()

# Evitar notación científica en los ejes
formatter = FuncFormatter(lambda x, _: f'{x:.0f}')
plt.gca().yaxis.set_major_formatter(formatter)

```

```
# Mostrar la gráfica
plt.grid(True)
plt.show()
```



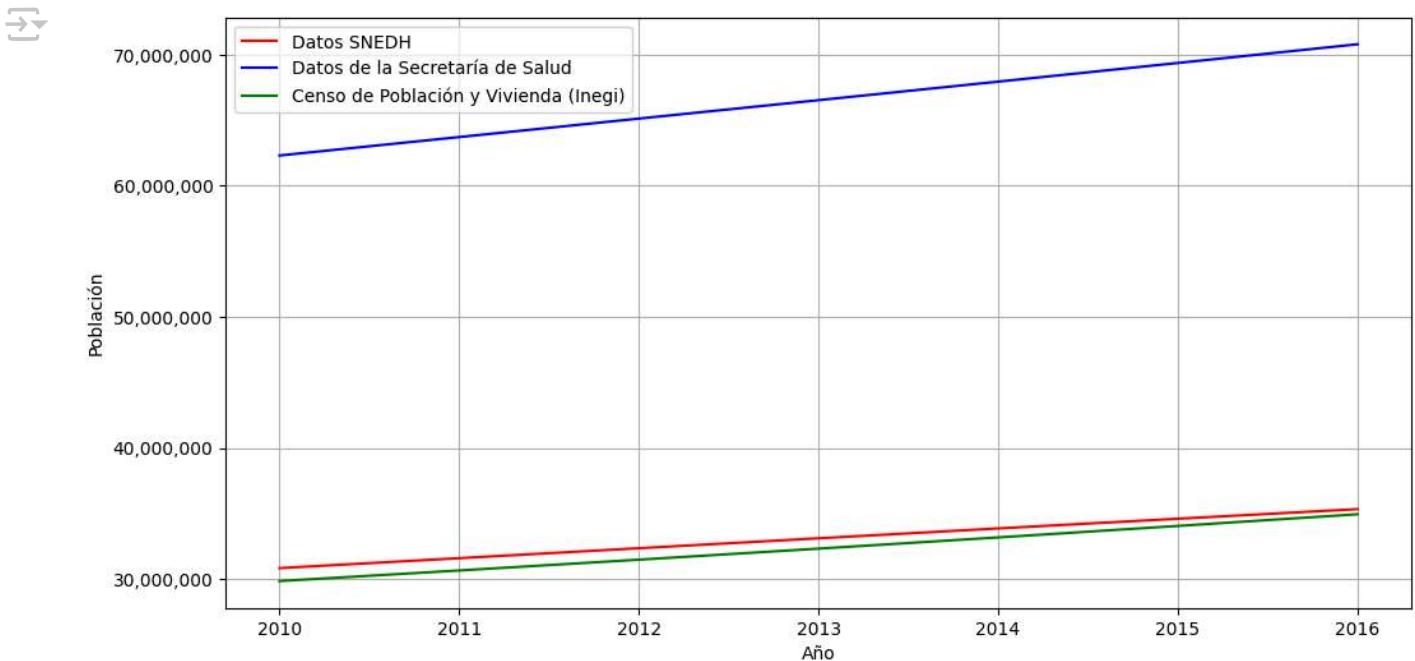
```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

# Verificar los años que abarca cada DataFrame
years_df_SNEDH = set(filas_estados_unidos_mexicanos['Año'])
years_df = set(df['Año'])
years_df_predictions = set(df_predictions['Año'])

# Encontrar los años comunes entre los tres DataFrames
common_years = sorted(years_df_SNEDH.intersection(years_df).intersection(years_df_predictions))

# Filtrar cada DataFrame para que solo contenga los años comunes
df_SNEDH_common = filas_estados_unidos_mexicanos[filas_estados_unidos_mexicanos['Año'].isin(common_years)]
df_common = df[df['Año'].isin(common_years)]
```

```
df_predictions_common = df_predictions[df_predictions['Año'].isin(common_years)]  
  
# Crear un nuevo DataFrame que contenga solo los años comunes y las columnas necesarias  
df_comparacion = pd.merge(df_SNEDH_common[['Año', 'Pfem']], df_common[['Año', 'Población de  
df_comparacion = pd.merge(df_comparacion, df_predictions_common[['Año', 'Población estimada']]  
  
# Renombrar las columnas para mayor claridad  
df_comparacion.columns = ["Año", "Pfem (Estados Unidos Mexicanos)", "Población de mujeres 25+ (df)"]  
  
# Graficar los resultados  
plt.figure(figsize=(12, 6))  
  
# Graficar cada serie  
plt.plot(df_comparacion["Año"], df_comparacion["Pfem (Estados Unidos Mexicanos)"], color="red")  
plt.plot(df_comparacion["Año"], df_comparacion["Población de mujeres 25+ (df)"], color="blue")  
plt.plot(df_comparacion["Año"], df_comparacion["Población estimada (df_predictions)"], color="green")  
  
# Añadir etiquetas y título  
plt.xlabel("Año")  
plt.ylabel("Población")  
plt.title("Comparación de intersección de la Población Femenina de 25 años o más en México")  
  
# Añadir leyenda  
plt.legend()  
  
# Evitar notación científica en los ejes  
formatter = FuncFormatter(lambda x, _: f'{x:,.0f}')  
plt.gca().yaxis.set_major_formatter(formatter)  
  
# Mostrar la gráfica  
plt.grid(True)  
plt.show()  
  
# Mostrar el DataFrame comparado  
print(df_comparacion)
```



	Año	Pfem (Estados Unidos Mexicanos)	Población de mujeres 25+ (df)	\
0	2010	30844960	62304782	
1	2011	31604691	63710053	
2	2012	32362693	65116779	
3	2013	33119331	66521317	
4	2014	33873575	67935405	
5	2015	34611681	69358429	
6	2016	35344978	70783734	

Población estimada (df\_predictions)

0	29859742
1	30666981
2	31490697
3	32331058
4	33188235
5	34062395
6	34953708

```
# Mostrar el DataFrame final
df_comparacion
```

	Año	Pfem (Estados Unidos Mexicanos)	Población de mujeres 25+ (df)	Población estimada (df_predictions)
0	2010	30844960	62304782	29859742
1	2011	31604691	63710053	30666981
2	2012	32362693	65116779	31490697
3	2013	33119331	66521317	32331058
4	2014	33873575	67935405	33188235
5	2015	34611681	69358429	34062395
6	2016	35344978	70783734	34953708

```
# Filtrar los datos entre 2010 y 2016 en el DataFrame df_predictions
df_filtered = df[(df['Año'] >= 2010) & (df['Año'] <= 2016)]
```

```
# Mostrar el resultado filtrado
```

```
df_filtered
```

	Año	Defunciones por cáncer de mama en mujeres de 25 años o más	Población de mujeres de 25 años o más	Tasa de mortalidad por cáncer de mama
20	2010	10074	62304782	16
21	2011	10414	63710053	16
22	2012	11168	65116779	17
23	2013	11052	66521317	17
24	2014	11948	67935405	18
25	2015	12552	69358429	18

```
import pandas as pd
ruta_guardado = '/content/drive/MyDrive/Cancer de mama/Tasa de mortalidad de 1990 - 2013 en df = pd.read_csv(ruta_guardado)
df
```



Año	Defunciones por cáncer de mama en mujeres de 25 años o más	Población de mujeres de 25 años o más	Tasa de mortalidad por cáncer de mama
0 1990	4380	35852664	12
1 1991	4668	37007840	13
2 1992	5036	38193064	13
3 1993	5352	39392350	14
4 1994	5478	40623620	13
5 1995	5960	41890024	14
6 1996	6130	43182190	14
7 1997	6362	44490494	14
8 1998	6692	45816222	15
9 1999	6764	47132266	14
10 2000	6838	48432714	14
11 2001	7126	49756872	14
12 2002	7644	51131594	15
13 2003	7722	52529180	15
14 2004	8300	53906692	15
15 2005	8410	55250834	15
16 2006	8880	56601780	16
17 2007	9162	57999644	16
18 2008	9604	59435352	16
19 2009	9786	60882268	16
20 2010	10074	62304782	16
21 2011	10414	63710053	16
22 2012	11168	65116779	17
23 2013	11052	66521317	17
24 2014	11948	67935405	18
25 2015	12552	69358429	18

df\_predictions

	Año	Población estimada
0	1895	2948882
1	1896	2988943
2	1897	3026038
3	1898	3060337
4	1899	3092009
...	...	...
124	2019	37732257
125	2020	38693873
126	2021	39673487
127	2022	40671268
128	2023	41687386

129 rows × 2 columns

```
# Filtrar el DataFrame para que incluya los años entre 1990 y 2016 (inclusive)
df_1990_onwards = df_predictions[(df_predictions['Año'] >= 1990) & (df_predictions['Año'] <= 2016)]

# Mostrar el DataFrame resultante
df_1990_onwards
```

	Año	Población estimada
95	1990	16914722
96	1991	17427929
97	1992	17954232
98	1993	18493799
99	1994	19046800
100	1995	19613404
101	1996	20193779
102	1997	20788095
103	1998	21396521
104	1999	22019226
105	2000	22656379
106	2001	23308149
107	2002	23974705
108	2003	24656217
109	2004	25352852
110	2005	26064781
111	2006	26792172
112	2007	27535195
113	2008	28294018
114	2009	29068811
115	2010	29859742
116	2011	30666981
117	2012	31490697
118	2013	32331058
119	2014	33188235
120	2015	34062395
121	2016	34953708

```
# Reindexa el DataFrame y elimina el índice anterior  
df_1990_onwards = df_1990_onwards.reset_index(drop=True)  
df_1990_onwards
```

	Año	Población estimada
0	1990	16914722
1	1991	17427929
2	1992	17954232
3	1993	18493799
4	1994	19046800
5	1995	19613404
6	1996	20193779
7	1997	20788095
8	1998	21396521
9	1999	22019226
10	2000	22656379
11	2001	23308149
12	2002	23974705
13	2003	24656217
14	2004	25352852
15	2005	26064781
16	2006	26792172
17	2007	27535195
18	2008	28294018
19	2009	29068811
20	2010	29859742
21	2011	30666981
22	2012	31490697
23	2013	32331058
24	2014	33188235
25	2015	34062395
26	2016	34953708

```
# Asegúrate de que los índices de ambos DataFrames estén alineados correctamente
df_1990_onwards['poblacion'] = df.loc[df_1990_onwards.index, 'Población de mujeres de 25 años']
df_1990_onwards
```

	Año	Población estimada	poblacion
0	1990	16914722	35852664
1	1991	17427929	37007840
2	1992	17954232	38193064
3	1993	18493799	39392350
4	1994	19046800	40623620
5	1995	19613404	41890024
6	1996	20193779	43182190
7	1997	20788095	44490494
8	1998	21396521	45816222
9	1999	22019226	47132266
10	2000	22656379	48432714
11	2001	23308149	49756872
12	2002	23974705	51131594
13	2003	24656217	52529180
14	2004	25352852	53906692
15	2005	26064781	55250834
16	2006	26792172	56601780
17	2007	27535195	57999644
18	2008	28294018	59435352
19	2009	29068811	60882268
20	2010	29859742	62304782
21	2011	30666981	63710053
22	2012	31490697	65116779
23	2013	32331058	66521317
24	2014	33188235	67935405
25	2015	34062395	69358429
26	2016	34953708	70783734

```
# Calcular el porcentaje de aumento entre "Población estimada" y "población"
df_1990_onwards['Porcentaje aumento'] = df_1990_onwards.apply(
    lambda row: ((row['poblacion'] - row['Población estimada']) / row['Población estimada']))
```

)

```
# Calcular la diferencia entre "poblacion" y "Población estimada"
df_1990_onwards['Diferencia'] = df_1990_onwards['poblacion'] - df_1990_onwards['Población es']
df_1990_onwards
```

	Año	Población estimada	poblacion	Porcentaje aumento	Diferencia
0	1990	16914722	35852664	112	18937942
1	1991	17427929	37007840	112	19579911
2	1992	17954232	38193064	113	20238832
3	1993	18493799	39392350	113	20898551
4	1994	19046800	40623620	113	21576820
5	1995	19613404	41890024	114	22276620
6	1996	20193779	43182190	114	22988411
7	1997	20788095	44490494	114	23702399
8	1998	21396521	45816222	114	24419701
9	1999	22019226	47132266	114	25113040
10	2000	22656379	48432714	114	25776335
11	2001	23308149	49756872	113	26448723
12	2002	23974705	51131594	113	27156889
13	2003	24656217	52529180	113	27872963
14	2004	25352852	53906692	113	28553840
15	2005	26064781	55250834	112	29186053
16	2006	26792172	56601780	111	29809608
17	2007	27535195	57999644	111	30464449
18	2008	28294018	59435352	110	31141334
19	2009	29068811	60882268	109	31813457
20	2010	29859742	62304782	109	32445040
21	2011	30666981	63710053	108	33043072
22	2012	31490697	65116779	107	33626082
23	2013	32331058	66521317	106	34190259
24	2014	33188235	67935405	105	34747170
25	2015	34062395	69358429	104	35296034
26	2016	34953708	70783734	103	35830026

```
df_1990_onwards.columns = ['Año', 'Población INEGI', 'Población Secretaría de salud', 'Diferencia']
df_1990_onwards
```



	Año	Población INEGI	Población Secretaría de salud	Diferencia	Porcentaje aumento
0	1990	16914722	35852664	18937942	112
1	1991	17427929	37007840	19579911	112
2	1992	17954232	38193064	20238832	113
3	1993	18493799	39392350	20898551	113
4	1994	19046800	40623620	21576820	113
5	1995	19613404	41890024	22276620	114
6	1996	20193779	43182190	22988411	114
7	1997	20788095	44490494	23702399	114
8	1998	21396521	45816222	24419701	114
9	1999	22019226	47132266	25113040	114
10	2000	22656379	48432714	25776335	114
11	2001	23308149	49756872	26448723	113
12	2002	23974705	51131594	27156889	113
13	2003	24656217	52529180	27872963	113
14	2004	25352852	53906692	28553840	113
15	2005	26064781	55250834	29186053	112
16	2006	26792172	56601780	29809608	111
17	2007	27535195	57999644	30464449	111
18	2008	28294018	59435352	31141334	110
19	2009	29068811	60882268	31813457	109
20	2010	29859742	62304782	32445040	109
21	2011	30666981	63710053	33043072	108
22	2012	31490697	65116779	33626082	107
23	2013	32331058	66521317	34190259	106
24	2014	33188235	67935405	34747170	105
25	2015	34062395	69358429	35296034	104
26	2016	34953708	70783734	35830026	103

```
# Intercambiar las últimas dos columnas
df_1990_onwards[['Porcentaje aumento', 'Diferencia']] = df_1990_onwards[['Diferencia', 'Porcentaje aumento']]
df_1990_onwards
```

	Año	Población INEGI	Población Secretaría de salud	Porcentaje aumento	Diferencia
0	1990	16914722	35852664	18937942	112
1	1991	17427929	37007840	19579911	112
2	1992	17954232	38193064	20238832	113
3	1993	18493799	39392350	20898551	113
4	1994	19046800	40623620	21576820	113
5	1995	19613404	41890024	22276620	114
6	1996	20193779	43182190	22988411	114
7	1997	20788095	44490494	23702399	114
8	1998	21396521	45816222	24419701	114
9	1999	22019226	47132266	25113040	114
10	2000	22656379	48432714	25776335	114
11	2001	23308149	49756872	26448723	113
12	2002	23974705	51131594	27156889	113
13	2003	24656217	52529180	27872963	113
14	2004	25352852	53906692	28553840	113
15	2005	26064781	55250834	29186053	112
16	2006	26792172	56601780	29809608	111
17	2007	27535195	57999644	30464449	111
18	2008	28294018	59435352	31141334	110
19	2009	29068811	60882268	31813457	109
20	2010	29859742	62304782	32445040	109
21	2011	30666981	63710053	33043072	108
22	2012	31490697	65116779	33626082	107
23	2013	32331058	66521317	34190259	106
24	2014	33188235	67935405	34747170	105
25	2015	34062395	69358429	35296034	104
26	2016	34953708	70783734	35830026	103

df\_1990\_onwards

	Año	Población INEGI	Población Secretaría de salud	Diferencia	Porcentaje aumento
0	1990	16914722	35852664	18937942	112
1	1991	17427929	37007840	19579911	112
2	1992	17954232	38193064	20238832	113
3	1993	18493799	39392350	20898551	113
4	1994	19046800	40623620	21576820	113
5	1995	19613404	41890024	22276620	114
6	1996	20193779	43182190	22988411	114
7	1997	20788095	44490494	23702399	114
8	1998	21396521	45816222	24419701	114
9	1999	22019226	47132266	25113040	114
10	2000	22656379	48432714	25776335	114
11	2001	23308149	49756872	26448723	113
12	2002	23974705	51131594	27156889	113
13	2003	24656217	52529180	27872963	113
14	2004	25352852	53906692	28553840	113
15	2005	26064781	55250834	29186053	112
16	2006	26792172	56601780	29809608	111
17	2007	27535195	57999644	30464449	111
18	2008	28294018	59435352	31141334	110
19	2009	29068811	60882268	31813457	109
20	2010	29859742	62304782	32445040	109
21	2011	30666981	63710053	33043072	108
22	2012	31490697	65116779	33626082	107
23	2013	32331058	66521317	34190259	106
24	2014	33188235	67935405	34747170	105
25	2015	34062395	69358429	35296034	104
26	2016	34953708	70783734	35830026	103

```
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker

# Crear una figura y ejes para graficar
fig, ax1 = plt.subplots(figsize=(10, 6))

# Graficar la Población INEGI
ax1.plot(df_1990_onwards['Año'], df_1990_onwards['Población INEGI'], color='tab:blue', label='Población INEGI')
ax1.set_xlabel('Año')
ax1.set_ylabel('Población INEGI', color='tab:blue')
ax1.tick_params(axis='y', labelcolor='tab:blue')

# Graficar la Población Secretaría de Salud
ax1.plot(df_1990_onwards['Año'], df_1990_onwards['Población Secretaría de salud'], color='tab:green', label='Población Secretaría de Salud')
ax1.set_ylabel('Población Secretaría de Salud', color='tab:green')
ax1.tick_params(axis='y', labelcolor='tab:green')

# Crear un segundo eje para graficar el porcentaje de aumento
ax2 = ax1.twinx() # Crea un eje adicional con el mismo eje x
ax2.plot(df_1990_onwards['Año'], df_1990_onwards['Porcentaje aumento'], color='tab:red', label='Porcentaje aumento')
ax2.set_ylabel('Porcentaje aumento', color='tab:red')
ax2.tick_params(axis='y', labelcolor='tab:red')

# Establecer los límites del eje y del porcentaje de aumento entre 0 y 120
ax2.set_ylim(0, 120)

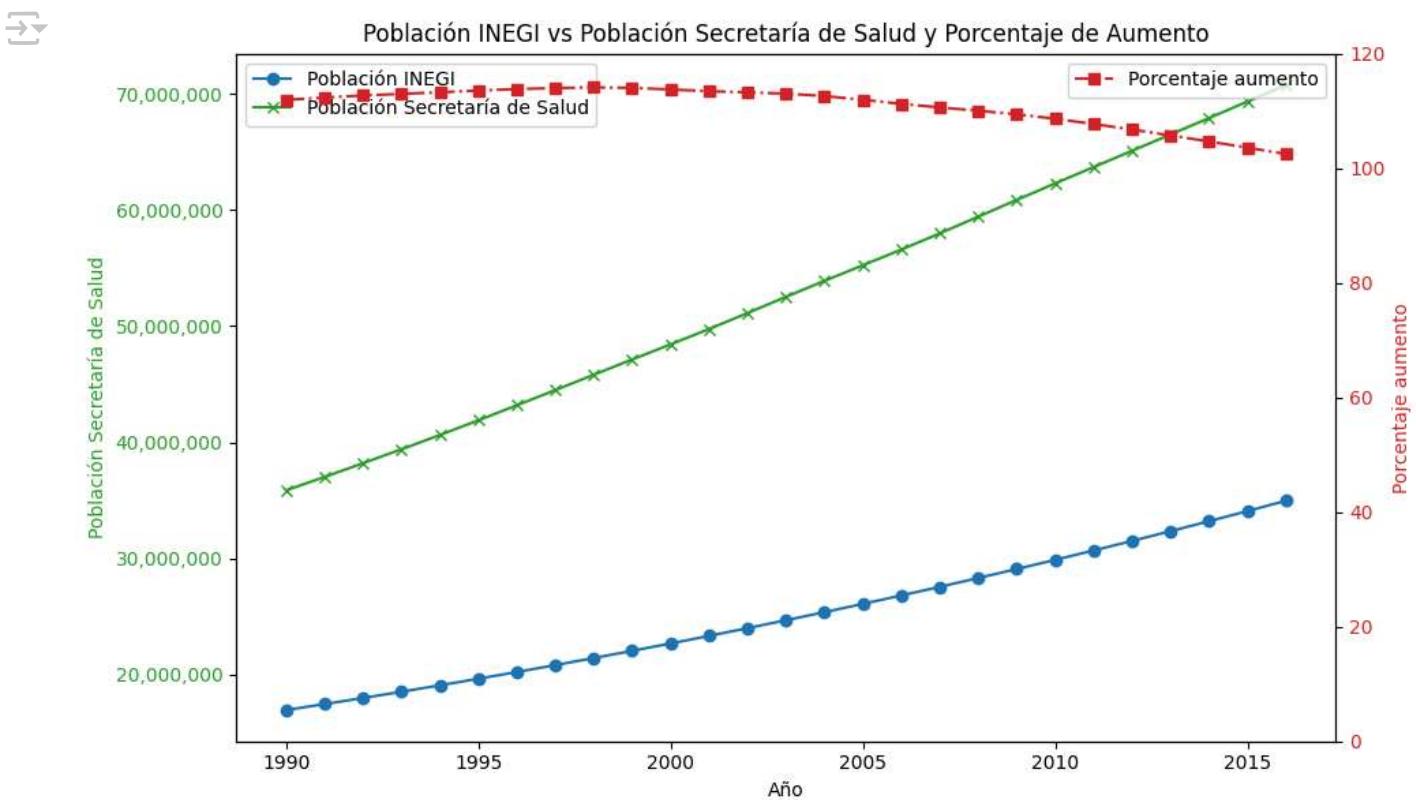
# Desactivar notación científica
ax1.yaxis.set_major_formatter(ticker.FuncFormatter(lambda x, _: f'{int(x)}')) # Población INEGI
ax2.yaxis.set_major_formatter(ticker.FuncFormatter(lambda x, _: f'{int(x)},.0f}')) # Porcentaje aumento

# Agregar título y leyenda
plt.title('Población INEGI vs Población Secretaría de Salud y Porcentaje de Aumento')

# Mostrar leyenda
ax1.legend(loc='upper left') # Leyenda para las poblaciones
ax2.legend(loc='upper right') # Leyenda para el porcentaje de aumento

fig.tight_layout() # Ajustar el layout para que no se solapen los ejes

# Mostrar el gráfico
plt.show()
```



```

import matplotlib.pyplot as plt

# Crear una figura y ejes para graficar
fig, ax = plt.subplots(figsize=(10, 6))

# Definir el ancho de las barras
bar_width = 0.35

# Definir las posiciones en el eje X para las barras
index = range(len(df_1990_onwards['Año']))

# Graficar las barras para la Población INEGI
ax.bar(index, df_1990_onwards['Población INEGI'], bar_width, label='Población INEGI', color='blue')

# Graficar las barras para la Población Secretaría de Salud, desplazada por un ancho de barr
ax.bar([i + bar_width for i in index], df_1990_onwards['Población Secretaría de salud'], bar_width, label='Población Secretaría de Salud', color='green')

# Etiquetas de los ejes
ax.set_xlabel('Año')
ax.set_ylabel('Población')
ax.set_title('Población INEGI vs Población Secretaría de Salud y Porcentaje de Aumento')

# Mostrar la leyenda
ax.legend()

# Mostrar el gráfico
plt.show()

```

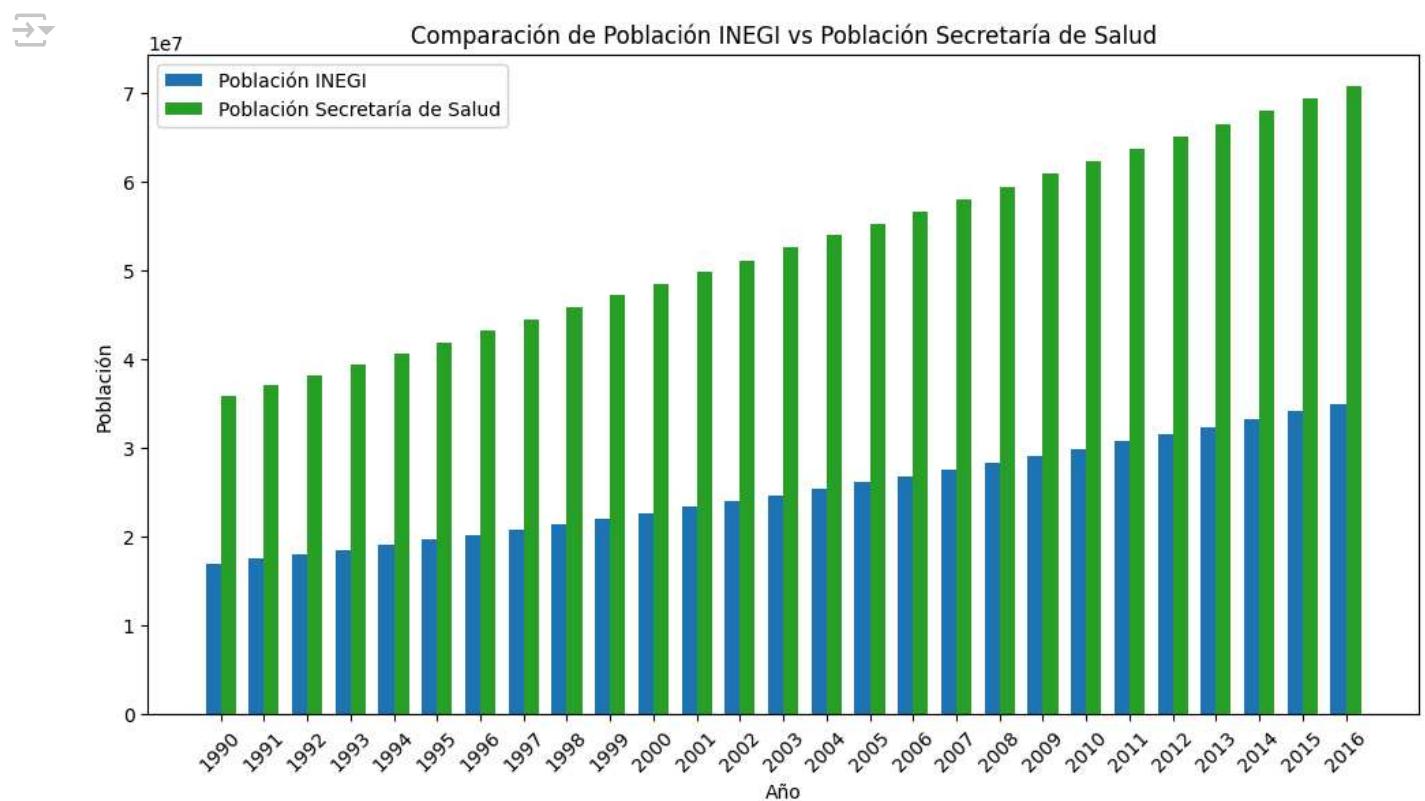
```
ax.set_xlabel('Año')
ax.set_ylabel('Población')

# Ajustar las posiciones del eje X
ax.set_xticks([i + bar_width / 2 for i in index]) # Centrar las etiquetas de los años
ax.set_xticklabels(df_1990_onwards['Año'], rotation=45)

# Título
ax.set_title('Comparación de Población INEGI vs Población Secretaría de Salud')

# Leyenda
ax.legend()

# Mostrar gráfico
plt.tight_layout() # Ajustar el layout para que no se solapen los elementos
plt.show()
```



```
df_predictions
```

	Año	Población estimada
0	1895	2948882
1	1896	2988943
2	1897	3026038
3	1898	3060337
4	1899	3092009
...	...	...
124	2019	37732257
125	2020	38693873
126	2021	39673487
127	2022	40671268
128	2023	41687386

129 rows × 2 columns

```
df_filtered = df_predictions[df_predictions['Año'] >= 2000].reset_index(drop=True)  
df_filtered
```