main

May 28, 2025

1 R2-A5-S16 Modelo de regresión o clasificación

1.1 Estudiante: José Miguel Méndez Martín

1.1.1 Ejercicio 1:

[5]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# --- 1. Cargar los datos ---
# Cargar los datos de esperanza de vida
try:
    # Las primeras dos líneas de tu ejemplo son metadatos, la tercera es el␣
 ↪encabezado
    df_life_exp_raw = pd.read_csv('../csv/API_SP.DYN.LE00.
 ↪IN_DS2_es_csv_v2_86998.csv')
except FileNotFoundError:
    print("Error: 'esperanza_vida_raw.csv' no encontrado. Asegúrate de que el␣
 ↪archivo esté en la misma carpeta o proporciona la ruta completa.")
    exit()

# Cargar los metadatos de los países (Región, Grupo de Ingresos)
try:
    df_country_meta = pd.read_csv('../csv/Metadata_Country_API_SP.DYN.LE00.
 ↪IN_DS2_es_csv_v2_86998.csv')
    # Limpiar posibles espacios en blanco en los nombres de las columnas
    df_country_meta.columns = df_country_meta.columns.str.strip()
except FileNotFoundError:
    print("Error: 'country_metadata.csv' no encontrado. Asegúrate de que el␣
 ↪archivo esté en la misma carpeta o proporciona la ruta completa.")
    exit()

print("--- Primeras filas de Esperanza de Vida (raw) ---")
print(df_life_exp_raw.head())
print("\n--- Primeras filas de Metadatos de Países ---")
print(df_country_meta.head())
```

```
# --- 2. Transformar datos de Esperanza de Vida a formato largo ---
id_vars_life_exp = ['Country Name', 'Country Code', 'Indicator Name',␣
 ↪'Indicator Code']
value_vars_life_exp = [col for col in df_life_exp_raw.columns if col not in␣
 ↪id_vars_life_exp and col.isdigit()]

df_life_exp_long = pd.melt(df_life_exp_raw,
                           id_vars=id_vars_life_exp,
                           value_vars=value_vars_life_exp,
                           var_name='Year',
                           value_name='Life_Expectancy')

df_life_exp_long['Year'] = pd.to_numeric(df_life_exp_long['Year'])
df_life_exp_long['Life_Expectancy'] = pd.
 ↪to_numeric(df_life_exp_long['Life_Expectancy'], errors='coerce')

print("\n--- Primeras filas de Esperanza de Vida (formato largo) ---")
print(df_life_exp_long.head())

# --- 3. Seleccionar un Año Específico ---
YEAR_TO_ANALYZE = 2021 # Elige un año reciente con datos
df_year_selected = df_life_exp_long[df_life_exp_long['Year'] ==␣
 ↪YEAR_TO_ANALYZE].copy()
df_year_selected = df_year_selected[['Country Name', 'Country Code',␣
 ↪'Life_Expectancy']] # Solo columnas necesarias

print(f"\n--- Datos de Esperanza de Vida para el año {YEAR_TO_ANALYZE} ---")
print(df_year_selected.head())

# --- 4. Limpiar df_country_meta y Fusionar ---
# Eliminar filas en df_country_meta donde 'Country Name' o 'Country Code' es␣
 ↪NaN, si las hay
df_country_meta.dropna(subset=['Country Name', 'Country Code'], inplace=True)
# Seleccionar columnas relevantes de metadatos
df_country_meta = df_country_meta[['Country Name', 'Country Code', 'Region',␣
 ↪'Income_Group']]

df_merged = pd.merge(df_year_selected, df_country_meta, on=['Country Name',␣
 ↪'Country Code'], how='left')

print("\n--- Datos Fusionados (Esperanza de Vida + Metadatos) ---")
print(df_merged.head())
df_merged.info()

# --- 5. Manejar Valores Faltantes en el DataFrame Fusionado ---
```

```
# Importante: Eliminar filas donde la Esperanza de Vida es NaN
df_merged.dropna(subset=['Life_Expectancy'], inplace=True)

# Para 'Region' e 'Income_Group', puedes rellenar NaNs con "Desconocido" o␣
 ↪eliminar esas filas
# Aquí los rellenaremos para no perder países si solo falta esta info categórica
df_merged['Region'].fillna('Desconocida', inplace=True)
df_merged['Income_Group'].fillna('Desconocido', inplace=True)

# Filtrar agregados (ej. donde Income_Group es 'Agregados' o Region está vacía␣
 ↪o es 'Desconocida' si eso identifica agregados)
# Basado en tus datos, parece que los agregados tienen 'Income_Group' vacío o␣
 ↪no especificado
# Los países con 'Country Code' como AFE, AFW, ARB también son agregados.
# Una forma más robusta sería tener una lista de códigos de agregados o filtrar␣
 ↪por 'Income_Group' no siendo uno de los grupos de ingreso de países.
# Por ahora, vamos a filtrar donde 'Income_Group' NO es 'Agregados' (si es que␣
 ↪existe esa categoría)
# Y donde 'Region' no sea 'Desconocida' si eso implica un agregado
# Y donde 'Country Code' no sea uno de los códigos de agregados conocidos (AFE,␣
 ↪AFW, ARB...)
known_aggregate_codes = ['AFE', 'AFW', 'ARB', 'CEA', 'CEB', 'ECS', 'EMU',␣
 ↪'EUU', 'FCS', 'HIC', 'HPC', 'IBD', 'IBT', 'IDA', 'IDX', 'LAC', 'LCN', 'LDC',␣
 ↪'LIC', 'LMC', 'LMY', 'LTE', 'MEA', 'MIC', 'MNA', 'NAC', 'OED', 'OSS', 'PRE',␣
 ↪'PSS', 'PST', 'SAS', 'SSA', 'SSF', 'SST', 'TEA', 'TEC', 'TLA', 'TMN', 'TSA',␣
 ↪'UMC', 'WLD'] # Lista incompleta, necesitas revisar los datos del Banco␣
 ↪Mundial
df_countries_only = df_merged[~df_merged['Country Code'].
 ↪isin(known_aggregate_codes)].copy()

# También filtrar por filas donde 'Income_Group' podría indicar un agregado (ej.
 ↪ si contiene la palabra 'aggregate')
if 'Income_Group' in df_countries_only.columns:
    df_countries_only = df_countries_only[~df_countries_only['Income_Group'].
 ↪str.contains('aggregate', case=False, na=False)]
    # Filtrar para asegurar que Income_Group sea uno de los válidos (ej.␣
 ↪'Ingreso alto', 'Países de ingreso bajo', etc.)
    valid_income_groups = ['Ingreso alto', 'Países de ingreso bajo', 'Países␣
 ↪de ingreso mediano bajo', 'Ingreso mediano alto'] # Ajusta según tus datos
    df_countries_only = df_countries_only[df_countries_only['Income_Group'].
 ↪isin(valid_income_groups)]


print(f"\n--- Datos Solo de Países (después de filtrar agregados) para␣
 ↪{YEAR_TO_ANALYZE} ---")
print(df_countries_only.head())
```

```
print(f"Número de países después de filtrar: {len(df_countries_only)}")

if df_countries_only.empty:
    print(f"No quedan datos de países después de filtrar para el año␣
 ↪{YEAR_TO_ANALYZE}. Revisa los filtros o el año.")
    exit()

# --- 6. ¡TU PRIMER OBJETO VISUAL! ---
# Por ejemplo, Esperanza de Vida promedio por Grupo de Ingresos

plt.figure(figsize=(10, 6))
sns.boxplot(x='Income_Group', y='Life_Expectancy', data=df_countries_only,
            order=['Países de ingreso bajo', 'Países de ingreso mediano bajo',␣
 ↪'Ingreso mediano alto', 'Ingreso alto']) # Ordenar categorías
plt.title(f'Esperanza de Vida por Grupo de Ingresos ({YEAR_TO_ANALYZE})')
plt.xlabel('Grupo de Ingresos')
plt.ylabel('Esperanza de Vida (años)')
plt.xticks(rotation=25, ha='right') # Rotar etiquetas para mejor lectura
plt.tight_layout() # Ajustar layout para que todo quepa
plt.show()

# --- SEGUNDO OBJETO VISUAL ---
# Esperanza de Vida promedio por Región
plt.figure(figsize=(12, 7))
sns.boxplot(x='Region', y='Life_Expectancy', data=df_countries_only)
plt.title(f'Esperanza de Vida por Región ({YEAR_TO_ANALYZE})')
plt.xlabel('Región')
plt.ylabel('Esperanza de Vida (años)')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# --- TERCER OBJETO VISUAL ---
# Histograma de Esperanza de Vida para los países seleccionados
plt.figure(figsize=(8, 6))
sns.histplot(df_countries_only['Life_Expectancy'], kde=True, bins=15)
plt.title(f'Distribución de Esperanza de Vida (Países, {YEAR_TO_ANALYZE})')
plt.xlabel('Esperanza de Vida (años)')
plt.ylabel('Frecuencia')
plt.show()

# --- CUARTO OBJETO VISUAL (Ejemplo) ---
# Países con mayor esperanza de vida
top_10_life_exp = df_countries_only.nlargest(10, 'Life_Expectancy')
plt.figure(figsize=(12, 7))
sns.barplot(x='Life_Expectancy', y='Country Name', data=top_10_life_exp,␣
 ↪palette='viridis')
```

```
plt.title(f'Top 10 Países por Esperanza de Vida ({YEAR_TO_ANALYZE})')
plt.xlabel('Esperanza de Vida (años)')
plt.ylabel('País')
plt.tight_layout()
plt.show()

# --- QUINTO OBJETO VISUAL (Ejemplo) ---
# Países con menor esperanza de vida
bottom_10_life_exp = df_countries_only.nsmallest(10, 'Life_Expectancy')
plt.figure(figsize=(12, 7))
sns.barplot(x='Life_Expectancy', y='Country Name', data=bottom_10_life_exp,␣
 ↪palette='rocket')
plt.title(f'10 Países con Menor Esperanza de Vida ({YEAR_TO_ANALYZE})')
plt.xlabel('Esperanza de Vida (años)')
plt.ylabel('País')
plt.tight_layout()
plt.show()

# Con esto ya tienes 5 visualizaciones y un DataFrame ('df_countries_only')
# que está listo para ser enriquecido con MÁS INDICADORES PREDICTORES
# para tu modelo de regresión.
```

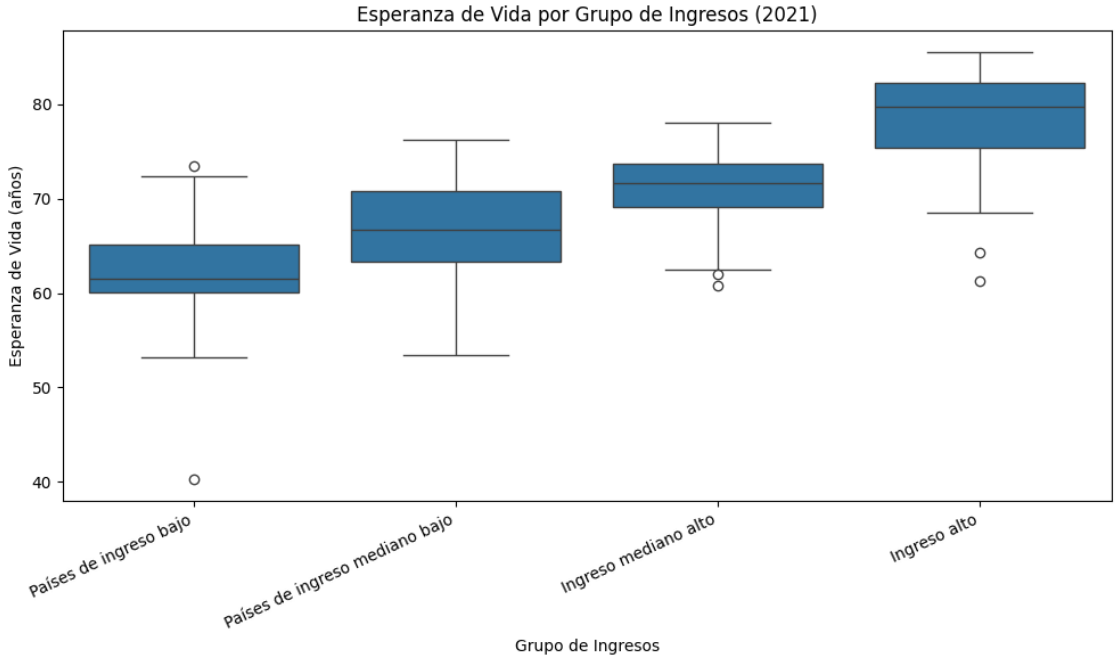
```
--- Primeras filas de Esperanza de Vida (raw) ---
  Country Name Country Code                             Indicator Name  \
0        Aruba          ABW  Esperanza de vida al nacer, total (años)
1          NaN          AFE  Esperanza de vida al nacer, total (años)
2   Afganistán          AFG  Esperanza de vida al nacer, total (años)
3          NaN          AFW  Esperanza de vida al nacer, total (años)
4       Angola          AGO  Esperanza de vida al nacer, total (años)

   Indicator Code       1960       1961       1962       1963       1964  \
0  SP.DYN.LE00.IN  64.049000  64.215000  64.602000  64.944000  65.303000
1  SP.DYN.LE00.IN  44.169257  44.468838  44.877890  45.160583  45.535695
2  SP.DYN.LE00.IN  32.799000  33.291000  33.757000  34.201000  34.673000
3  SP.DYN.LE00.IN  37.779636  38.058956  38.681792  38.936918  39.194580
4  SP.DYN.LE00.IN  37.933000  36.902000  37.168000  37.419000  37.704000

        1965  …       2016       2017       2018       2019       2020  \
0  65.615000  …  75.540000  75.620000  75.880000  76.019000  75.406000
1  45.770723  …  62.167981  62.591275  63.330691  63.857261  63.766484
2  35.124000  …  62.646000  62.406000  62.443000  62.941000  61.454000
3  39.479784  …  56.392452  56.626439  57.036976  57.149847  57.364425
4  37.968000  …  61.619000  62.122000  62.622000  63.051000  63.116000

        2021       2022       2023  2024  Unnamed: 69
0  73.655000  76.226000  76.353000   NaN          NaN
1  62.979999  64.487020  65.146291   NaN          NaN
2  60.417000  65.617000  66.035000   NaN          NaN
```

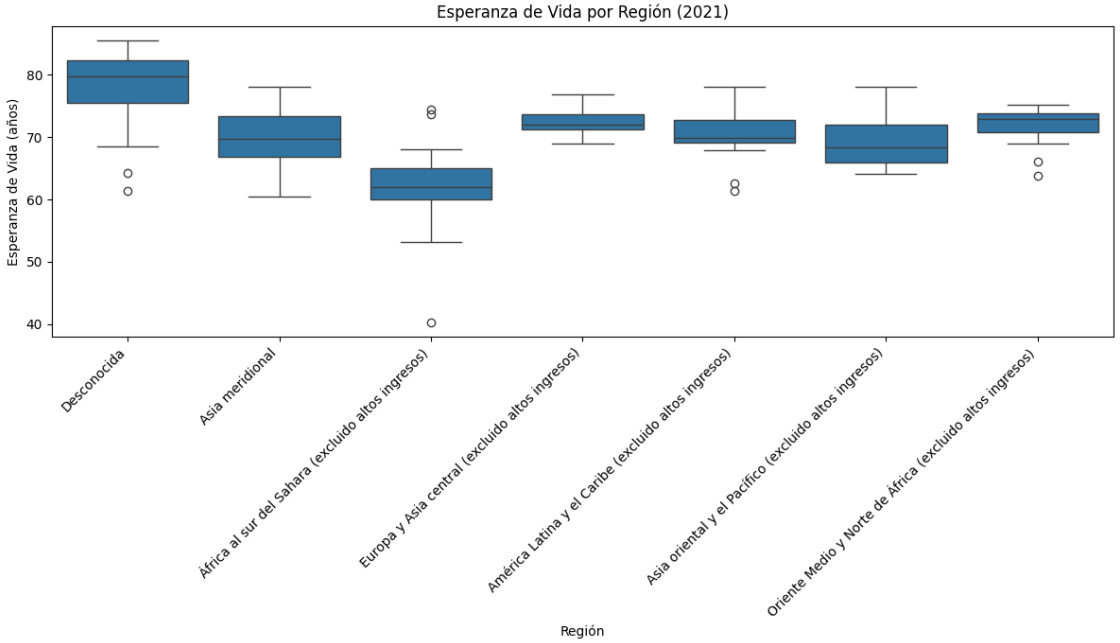
```
3  57.362572  57.987813  58.855722   NaN          NaN
4  62.958000  64.246000  64.617000   NaN          NaN

[5 rows x 70 columns]

--- Primeras filas de Metadatos de Países ---
  Country Name Country Code  \
0        Aruba          ABW
1          NaN          AFE
2   Afganistán          AFG
3          NaN          AFW
4       Angola          AGO

                                            Region  \
0                                              NaN
1                                              NaN
2                                  Asia meridional
3                                              NaN
4  África al sur del Sahara (excluido altos ingre…

                    Income_Group  Unnamed: 4
0                    Ingreso alto         NaN
1                       Agregados         NaN
2          Países de ingreso bajo         NaN
3                       Agregados         NaN
4  Países de ingreso mediano bajo         NaN

--- Primeras filas de Esperanza de Vida (formato largo) ---
  Country Name Country Code                             Indicator Name  \
0        Aruba          ABW  Esperanza de vida al nacer, total (años)
1          NaN          AFE  Esperanza de vida al nacer, total (años)
2   Afganistán          AFG  Esperanza de vida al nacer, total (años)
3          NaN          AFW  Esperanza de vida al nacer, total (años)
4       Angola          AGO  Esperanza de vida al nacer, total (años)

   Indicator Code  Year  Life_Expectancy
0  SP.DYN.LE00.IN  1960        64.049000
1  SP.DYN.LE00.IN  1960        44.169257
2  SP.DYN.LE00.IN  1960        32.799000
3  SP.DYN.LE00.IN  1960        37.779636
4  SP.DYN.LE00.IN  1960        37.933000

--- Datos de Esperanza de Vida para el año 2021 ---
      Country Name Country Code  Life_Expectancy
16226        Aruba          ABW        73.655000
16227          NaN          AFE        62.979999
16228   Afganistán          AFG        60.417000
16229          NaN          AFW        57.362572
```

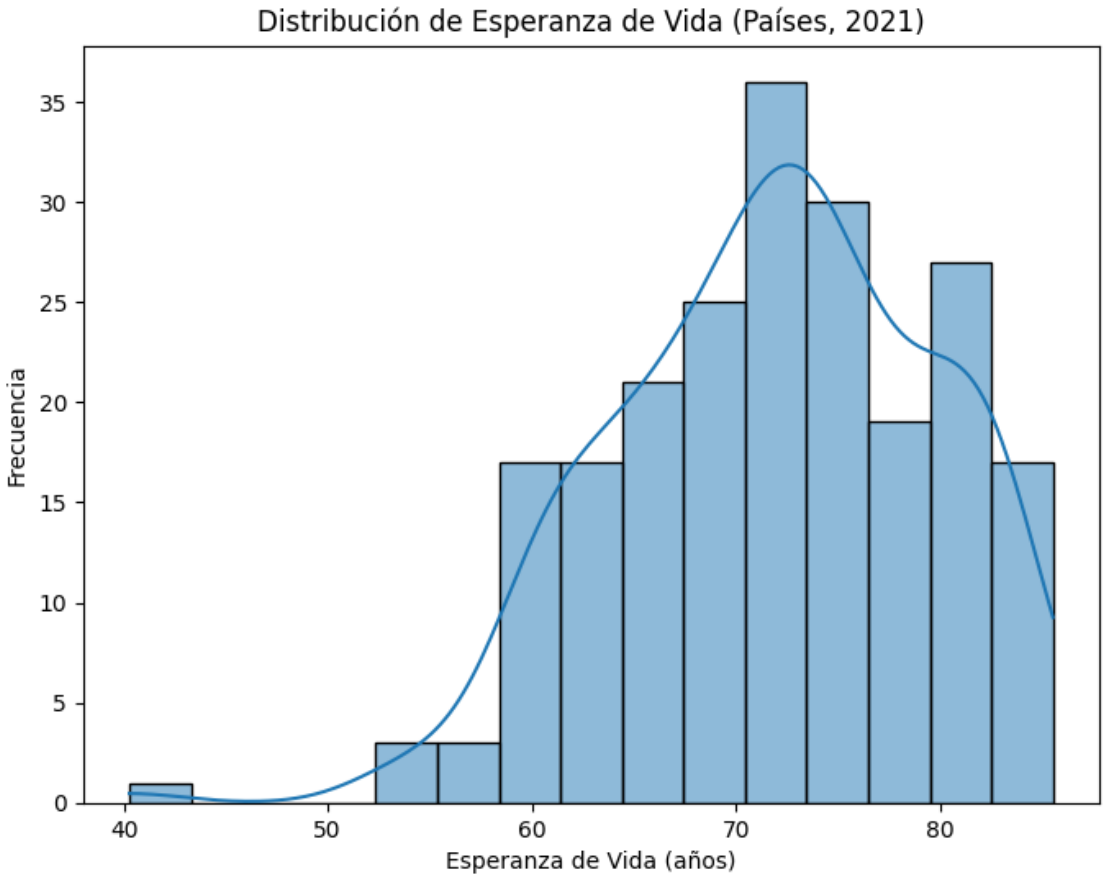
```
16230       Angola          AGO        62.958000

--- Datos Fusionados (Esperanza de Vida + Metadatos) ---
  Country Name Country Code  Life_Expectancy  \
0        Aruba          ABW        73.655000
1          NaN          AFE        62.979999
2   Afganistán          AFG        60.417000
3          NaN          AFW        57.362572
4       Angola          AGO        62.958000

                                            Region  \
0                                              NaN
1                                              NaN
2                                  Asia meridional
3                                              NaN
4  África al sur del Sahara (excluido altos ingre…

                    Income_Group
0                    Ingreso alto
1                             NaN
2          Países de ingreso bajo
3                             NaN
4  Países de ingreso mediano bajo
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 266 entries, 0 to 265
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Country Name     264 non-null    object
 1   Country Code     266 non-null    object
 2   Life_Expectancy  265 non-null    float64
 3   Region           131 non-null    object
 4   Income_Group     260 non-null    object
dtypes: float64(1), object(4)
memory usage: 10.5+ KB

--- Datos Solo de Países (después de filtrar agregados) para 2021 ---
  Country Name Country Code  Life_Expectancy  \
0        Aruba          ABW            73.655
2   Afganistán          AFG            60.417
4       Angola          AGO            62.958
5      Albania          ALB            76.844
6      Andorra          AND            82.331

                                            Region  \
0                                      Desconocida
2                                  Asia meridional
4  África al sur del Sahara (excluido altos ingre…
```

```
5    Europa y Asia central (excluido altos ingresos)
6                                        Desconocida

                    Income_Group
0                    Ingreso alto
2          Países de ingreso bajo
4  Países de ingreso mediano bajo
5            Ingreso mediano alto
6                    Ingreso alto
Número de países después de filtrar: 216
```

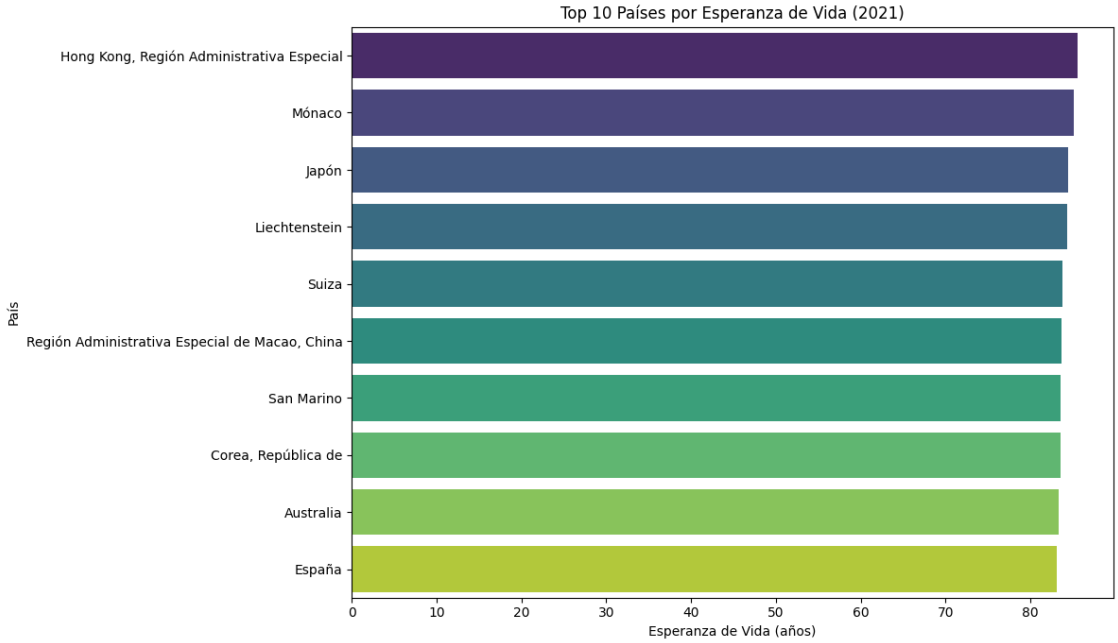
```
C:\Users\josem\AppData\Local\Temp\ipykernel_6608\2155049644.py:71:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  df_merged['Region'].fillna('Desconocida', inplace=True)
C:\Users\josem\AppData\Local\Temp\ipykernel_6608\2155049644.py:72:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  df_merged['Income_Group'].fillna('Desconocido', inplace=True)
```


Esperanza de Vida por Grupo de Ingresos (2021)
Esperanza de Vida (años)
80
70
60
50
40
Países de ingreso bajo
Países de ingreso mediano bajo
Ingreso mediano alto
Ingreso alto
Grupo de Ingresos

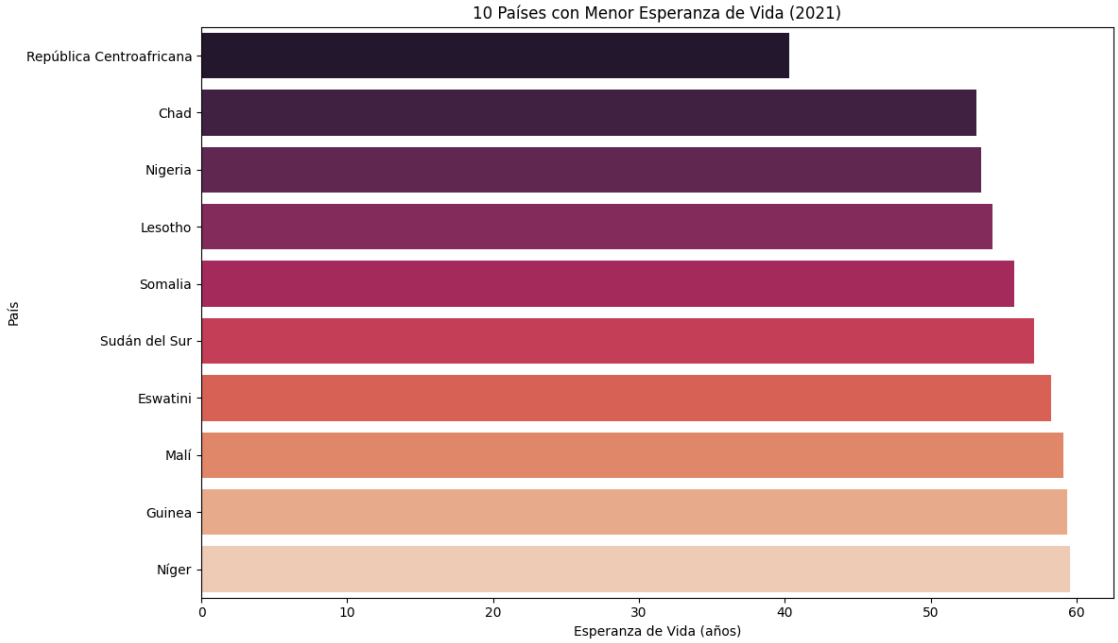

Esperanza de Vida por Región (2021)
Esperanza de Vida (años)
80
70
60
50
40
Desconocida
Asia meridional
África al sur del Sahara (excluido altos ingresos)
Europa y Asia central (excluido altos ingresos)
América Latina y el Caribe (excluido altos ingresos)
Asia oriental y el Pacífico (excluido altos ingresos)
Oriente Medio y Norte de África (excluido altos ingresos)
Región




```
C:\Users\josem\AppData\Local\Temp\ipykernel_6608\2155049644.py:137:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x='Life_Expectancy', y='Country Name', data=top_10_life_exp,
palette='viridis')
```




```
C:\Users\josem\AppData\Local\Temp\ipykernel_6608\2155049644.py:148:
FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x='Life_Expectancy', y='Country Name', data=bottom_10_life_exp,
palette='rocket')
```


10 Países con Menor Esperanza de Vida (2021)
República Centroafricana
Chad
Nigeria
Lesotho
Somalia
Sudán del Sur
Eswatini
Malí
Guinea
Níger
País
0
10
20
30
40
50
60
Esperanza de Vida (años)