Este código implementa una aplicación interactiva para analizar y predecir la esperanza de vida por país utilizando un modelo de **Random Forest** y otras técnicas estadísticas. A continuación, te hago un resumen detallado de lo que hace cada sección del código:

**1. Instalación de librerías**

python

CopiarEditar

!pip install gradio seaborn scikit-learn plotly --quiet

!pip install pandas==2.2.2 --quiet

Estas líneas instalan las librerías necesarias para ejecutar el código:

* **Gradio** para la interfaz de usuario interactiva.
* **Seaborn** y **Matplotlib** para la visualización de datos.
* **Scikit-learn** para los modelos de machine learning.
* **Plotly** para la visualización interactiva (mapas, gráficos).
* **Pandas** para la manipulación de datos.

**2. Importación de librerías**

python

CopiarEditar

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import gradio as gr

import plotly.express as px

from sklearn.ensemble import RandomForestRegressor

from sklearn.model\_selection import train\_test\_split, GridSearchCV

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

import numpy as np

Aquí se importan todas las librerías necesarias:

* **Pandas** para el manejo de los datos.
* **Matplotlib** y **Seaborn** para los gráficos estáticos.
* **Gradio** para crear la interfaz interactiva.
* **Plotly** para los mapas interactivos.
* **Scikit-learn** para crear y evaluar el modelo de Random Forest.
* **NumPy** para operaciones matemáticas.

**3. Carga de datos**

python

CopiarEditar

url = 'https://raw.githubusercontent.com/ramapu12/trabajoFinal/59399a459ba091dfd304faa2e1bdd509b56d5477/Life%20Expectancy%20Data.csv'

df\_completo = pd.read\_csv(url)

df\_completo.columns = df\_completo.columns.str.strip().str.replace(r'\s+', '\_', regex=True)

Se carga un archivo CSV desde una URL que contiene los datos de esperanza de vida por país, y se ajustan los nombres de las columnas para que no tengan espacios, facilitando su manejo.

**4. Filtrado de columnas**

python

CopiarEditar

variables\_modelo = [

'Life\_expectancy', 'Year', 'Country', 'Schooling',

'Income\_composition\_of\_resources', 'GDP', 'Alcohol',

'BMI', 'Diphtheria', 'HIV/AIDS'

]

df = df\_completo[variables\_modelo].dropna()

Se seleccionan las columnas relevantes para el modelo y se eliminan las filas con valores faltantes.

**5. Entrenamiento del modelo de Random Forest**

python

CopiarEditar

X = df.drop(columns=['Life\_expectancy', 'Country'])

y = df['Life\_expectancy']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

param\_grid = {

'n\_estimators': [50, 100, 200],

'max\_depth': [10, 20, 30, None],

'min\_samples\_split': [2, 5, 10]

}

grid\_search = GridSearchCV(RandomForestRegressor(random\_state=42), param\_grid, cv=3, n\_jobs=-1, verbose=2)

grid\_search.fit(X\_train, y\_train)

El modelo **Random Forest** se entrena con las variables seleccionadas. Se realiza un **GridSearchCV** para encontrar los mejores parámetros del modelo (como el número de árboles, la profundidad máxima y el número mínimo de muestras para dividir un nodo).

**6. Evaluación del modelo**

python

CopiarEditar

y\_pred = rf\_ajustado.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred))

r2 = r2\_score(y\_test, y\_pred)

Se evalúa el rendimiento del modelo utilizando las métricas **MAE** (error absoluto medio), **RMSE** (raíz del error cuadrático medio) y **R²** (coeficiente de determinación).

**7. Proyección de variables mediante regresión lineal**

python

CopiarEditar

def proyectar\_variable(df\_pais, variable, anios\_futuros):

from sklearn.linear\_model import LinearRegression

datos = df\_pais[['Year', variable]].dropna()

if datos.shape[0] < 2:

return np.repeat(datos[variable].values[-1], anios\_futuros)

X = datos['Year'].values.reshape(-1, 1)

y = datos[variable].values

modelo = LinearRegression().fit(X, y)

futuros = np.arange(df\_pais['Year'].max() + 1, df\_pais['Year'].max() + 1 + anios\_futuros).reshape(-1, 1)

return modelo.predict(futuros)

Se define una función para proyectar el valor de una variable (por ejemplo, el **PIB** o el **alcohol**) para los próximos años mediante **regresión lineal**. Si hay pocos datos, se devuelve el valor más reciente.

**8. Predicción de esperanza de vida con Random Forest**

python

CopiarEditar

def predecir\_rf(pais, anios\_futuros):

datos\_pais = df[df['Country'] == pais]

entrada = datos\_pais.sort\_values(by='Year', ascending=False).iloc[0]

X\_input = entrada.drop(labels=['Life\_expectancy', 'Country'])

prediccion = rf\_ajustado.predict([X\_input])[0]

Se define una función para predecir la esperanza de vida para un país usando el modelo de **Random Forest**. La entrada es la última fila de datos del país seleccionado, y la salida es la predicción para el año más reciente.

**9. Visualización de resultados**

python

CopiarEditar

fig, ax = plt.subplots(figsize=(6, 4))

sns.barplot(x=[entrada['Life\_expectancy'], prediccion], y=['Actual', 'Predicho'], palette='Blues', ax=ax)

Se genera un gráfico de barras para comparar la esperanza de vida real con la predicción realizada por el modelo.

**10. Importancia de las características**

python

CopiarEditar

importances = rf\_ajustado.feature\_importances\_

Se obtiene la importancia de cada variable (característica) en el modelo. Esto indica qué variables tienen mayor impacto en la predicción de la esperanza de vida.

**11. Visualización de la importancia de las características**

python

CopiarEditar

fig\_importance, ax\_importance = plt.subplots(figsize=(8, 6))

sns.barplot(x='Importance', y='Feature', data=importance\_df, ax=ax\_importance, palette='viridis')

Se genera un gráfico de barras que muestra la importancia de cada variable en el modelo, lo que ayuda a entender qué factores influyen más en la predicción de la esperanza de vida.

**12. Proyección futura de la esperanza de vida**

python

CopiarEditar

future\_years = np.arange(datos\_pais['Year'].max() + 1, datos\_pais['Year'].max() + 1 + anios\_futuros)

future\_input = pd.DataFrame()

Se realiza una proyección futura de la esperanza de vida utilizando las variables proyectadas para los próximos años. Luego se crea un gráfico que muestra la evolución histórica y las predicciones futuras.

**13. Generación de gráficos globales (Top 10 y Bottom 10 países)**

python

CopiarEditar

def graficos\_globales():

mean\_life = df.groupby('Country', as\_index=False)['Life\_expectancy'].mean()

top = mean\_life.sort\_values(by='Life\_expectancy', ascending=False).head(10)

bottom = mean\_life.sort\_values(by='Life\_expectancy', ascending=True).head(10)

Se genera un gráfico de barras con los 10 países con mayor y menor esperanza de vida. Esto ayuda a visualizar las diferencias globales en esperanza de vida.

**14. Mapa interactivo de esperanza de vida**

python

CopiarEditar

def mapa\_esperanza():

promedio\_pais = df.groupby('Country', as\_index=False)['Life\_expectancy'].mean()

fig = px.choropleth(promedio\_pais, locations="Country", color="Life\_expectancy", hover\_name="Country")

Se genera un **mapa interactivo** que muestra la esperanza de vida media por país, facilitando la visualización global.

**15. Comparativa con el país con mayor esperanza de vida**

python

CopiarEditar

def comparativa\_con\_mejor\_pais(pais, anios\_futuros):

mejor\_pais = obtener\_pais\_mejor\_esperanza()

...

Esta función compara la esperanza de vida del país seleccionado con el país que tiene la mayor esperanza de vida. Se genera un gráfico de comparación de las proyecciones de ambos países.

**16. Interfaz de usuario con Gradio**

python

CopiarEditar

with gr.Blocks(theme=gr.themes.Soft()) as demo:

gr.Markdown("## Comparativa Global de Esperanza de Vida")

...

Se crea una interfaz gráfica con **Gradio**. El usuario puede seleccionar un país, visualizar gráficos comparativos, interactuar con un mapa, y obtener predicciones de la esperanza de vida futura mediante el modelo de **Random Forest**.

**Resumen de lo que hace el código:**

El código proporciona una aplicación interactiva que permite a los usuarios:

1. **Seleccionar un país** y ver la esperanza de vida histórica y proyectada.
2. Comparar el país seleccionado con el país con la mayor esperanza de vida.
3. Visualizar un **mapa interactivo** de la esperanza de vida global.
4. Ver las **importancias de las variables** utilizadas por el modelo.
5. Realizar proyecciones futuras de la esperanza de vida mediante un modelo de **Random Forest** y **regresión lineal**.