Modelo de predicción de pobreza y pobreza extrema según los ingresos mensuales

- Usando los datos de https://sitservicios.lapaz.bo/portal-estadistico/#/datos-estadisticos documento número 2.02.01 «Líneas de pobreza y pobreza extrema en bolivianos por persona al mes según área»
- Si no están ya instaladas, instalar las siguientes librerías de python:

```
pip install pandas matplotlib scikit-learn mplcursors numpy
```

- Crear un archivo .csv llamado "pobreza_por_area.csv" que contenga:

```
Gestión,Bolivia Rural,Bolivia Urbana,Municipio de La Paz,Bolivia Rural Extrema,Bolivia Urbana
Extrema,Municipio de La Paz Extrema
2011,510.4,683.6,668.3,290.9,360.3,371.6
2012,523.9,693.2,690.4,298.6,365.5,383.9
2013,542.3,733.5,747.0,309.1,387.1,415.3
2014,552.6,759.3,760.7,315.0,400.6,423.0
2015,550.6,760.4,782.7,313.8,401.3,435.2
2016,593.5,875.3,1010.7,320.6,435.4,496.4
2017,622.9,887.2,1034.4,341.5,434.3,499.7
2018,629.9,888.4,1035.5,345.2,431.5,492.4
2019,668.1,911.7,1063.8,381.1,449.3,514.6
2020,635.1,902.2,1061.1,344.4,434.6,502.1
2021,637.1,947.5,1065.1,346.4,453.7,500.5
```

- Crear un archivo del codigo en python, por ejemplo, "proy.py" que contenga:

```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
import numpy as np
import mplcursors  # Para la interactividad

# Cargar los datos
data = pd.read_csv('pobreza_por_area.csv')

# Seleccionar la variable independiente (año) y las dependientes (líneas de pobreza)
X = data[['Gestión']]  # Aseguramos que 'Gestión' sea una columna de un DataFrame
y_rural = data['Bolivia Rural']
y_urbana = data['Bolivia Urbana']
y_lapaz = data['Municipio de La Paz']

# Dividir los datos en entrenamiento y prueba
X_train, X_test, y_train_rural, y_test_rural = train_test_split(X, y_rural, test_size=0.2,
random_state=42)
_, _, y_train_urbana, y_test_urbana = train_test_split(X, y_urbana, test_size=0.2,
random_state=42)
_, _, y_train_lapaz, y_test_lapaz = train_test_split(X, y_lapaz, test_size=0.2, random_state=42)

# Crear y entrenar modelos para cada área
model_rural = LinearRegression()
model_rural.fit(X_train, y_train_rural)

model_urbana = LinearRegression()
model_urbana.fit(X_train, y_train_urbana)

model_lapaz = LinearRegression()
model_lapaz.fit(X_train, y_train_lapaz)

# Calcular R^2 para cada modelo
r2_rural = model_rural.score(X_test, y_test_rural)
r2_urbana = model_urbana.score(X_test, y_test_urbana)
```

```
r2_lapaz = model_lapaz.score(X_test, y_test_lapaz)

# Mostrar R^2 en consola
print(f"R² para Rural: {r2_rural:.3f}")
print(f"R² para Urbana: {r2_urbana:.3f}")
print(f"R² para La Paz: {r2_lapaz:.3f}")

# Predecir para los próximos 6 años (2022-2027)
future_years = pd.DataFrame({
    'Gestión': [2022, 2023, 2024, 2025, 2026, 2027]  # Usamos un DataFrame para asegurarnos de
que tiene nombre
})

pred_future_rural = model_rural.predict(future_years[['Gestión']])
pred_future_urbana = model_urbana.predict(future_years[['Gestión']])
pred_future_lapaz = model_lapaz.predict(future_years[['Gestión']])

# Generar una predicción sobre todo el eje X
years_full = pd.DataFrame({'Gestión': np.arange(2010, 2028)})  # DataFrame para todo el rango de
años
pred_full_rural = model_rural.predict(years_full[['Gestión']])
pred_full_urbana = model_urbana.predict(years_full[['Gestión']])
pred_full_lapaz = model_lapaz.predict(years_full[['Gestión']])

# Visualizar las predicciones y agregar la línea de predicción para los próximos años
plt.figure(figsize=(12, 8))

# Gráfica de los puntos reales
real_rural_points = plt.scatter(X, y_rural, color='blue', label='Real Rural')
real_urbana_points = plt.scatter(X, y_urbana, color='green', label='Real Urbana')
real_lapaz_points = plt.scatter(X, y_lapaz, color='red', label='Real La Paz')

# Gráfica de las predicciones sobre todo el eje X
plt.plot(years_full, pred_full_rural, color='cyan', linestyle='dashed', label='Predicción Rural')
plt.plot(years_full, pred_full_urbana, color='lime', linestyle='dashed', label='Predicción
Urbana')
plt.plot(years_full, pred_full_lapaz, color='orange', linestyle='dashed', label='Predicción La
Paz')

# Puntos para las predicciones futuras con líneas conectando los puntos
future_rural_points = plt.scatter(future_years[['Gestión']], pred_future_rural, color='cyan',
zorder=5)
future_urbana_points = plt.scatter(future_years[['Gestión']], pred_future_urbana, color='lime',
zorder=5)
future_lapaz_points = plt.scatter(future_years[['Gestión']], pred_future_lapaz, color='orange',
zorder=5)

# Conectar los puntos de las predicciones con líneas para cada área
plt.plot(future_years[['Gestión']], pred_future_rural, color='cyan', linestyle='-', linewidth=2,
zorder=4)
plt.plot(future_years[['Gestión']], pred_future_urbana, color='lime', linestyle='-', linewidth=2,
zorder=4)
plt.plot(future_years[['Gestión']], pred_future_lapaz, color='orange', linestyle='-',
linewidth=2, zorder=4)

# Mostrar texto solo cuando el ratón pasa sobre los puntos reales
mplcursors.cursor(real_rural_points, hover=True).connect("add", lambda sel:
sel.annotation.set_text(f"Año: {data.iloc[sel.index]['Gestión']}\nPobreza Rural:
{sel.target[1]:.2f}"))
mplcursors.cursor(real_urbana_points, hover=True).connect("add", lambda sel:
sel.annotation.set_text(f"Año: {data.iloc[sel.index]['Gestión']}\nPobreza Urbana:
```

```
{sel.target[1]:.2f}"))
mplcursors.cursor(real_lapaz_points, hover=True).connect("add", lambda sel:
sel.annotation.set_text(f"Año: {data.iloc[sel.index]['Gestión']}\nPobreza La Paz:
{sel.target[1]:.2f}"))

# Mostrar texto siempre en los puntos de predicción futura (con el año)
mplcursors.cursor(future_rural_points, hover=True).connect("add", lambda sel:
sel.annotation.set_text(f"Año: {future_years.iloc[sel.index]['Gestión']}\nPobreza Rural:
{sel.target[1]:.2f}"))
mplcursors.cursor(future_urbana_points, hover=True).connect("add", lambda sel:
sel.annotation.set_text(f"Año: {future_years.iloc[sel.index]['Gestión']}\nPobreza Urbana:
{sel.target[1]:.2f}"))
mplcursors.cursor(future_lapaz_points, hover=True).connect("add", lambda sel:
sel.annotation.set_text(f"Año: {future_years.iloc[sel.index]['Gestión']}\nPobreza La Paz:
{sel.target[1]:.2f}"))

# Agregar detalles como líneas de cuadrícula y etiquetas
plt.xlabel('Año')
plt.ylabel('Líneas de Pobreza')
plt.title('Predicción de Pobreza por Área')
plt.grid(True)  # Mostrar líneas de cuadrícula
plt.legend()

# Mostrar el valor de R^2 debajo de la gráfica
plt.figtext(0.5, -0.1, f'R² para Rural: {r2_rural:.3f} | R² para Urbana: {r2_urbana:.3f} | R²
para La Paz: {r2_lapaz:.3f}', ha='center', va='top', fontsize=12)

plt.tight_layout()

# Mostrar el gráfico
plt.show()

# Mostrar las predicciones para los próximos años
print("Predicciones para los próximos años (2022-2027):")
for year, rural, urbana, lapaz in zip(range(2022, 2028), pred_future_rural, pred_future_urbana,
pred_future_lapaz):
    print(f"Año {year} - Bolivia Rural: {rural:.2f}, Bolivia Urbana: {urbana:.2f}, Municipio de
La Paz: {lapaz:.2f}")
```

- Asegurarse de usar el interprete correcto de python para ejecutarlo

La línea de pobreza es un umbral que se utiliza para determinar si las personas u hogares tienen los recursos suficientes para satisfacer sus necesidades básicas, como alimentación, vivienda y servicios. Aquellos que viven por debajo de esta línea son considerados en situación de pobreza.

En este caso, los números indican cuántos recursos (en bolivianos) se requieren para superar la pobreza en cada área. Esto significa que si los ingresos mensuales de una persona están por debajo de la línea de pobraza, entonces esa persona se consideraría en situación de pobreza