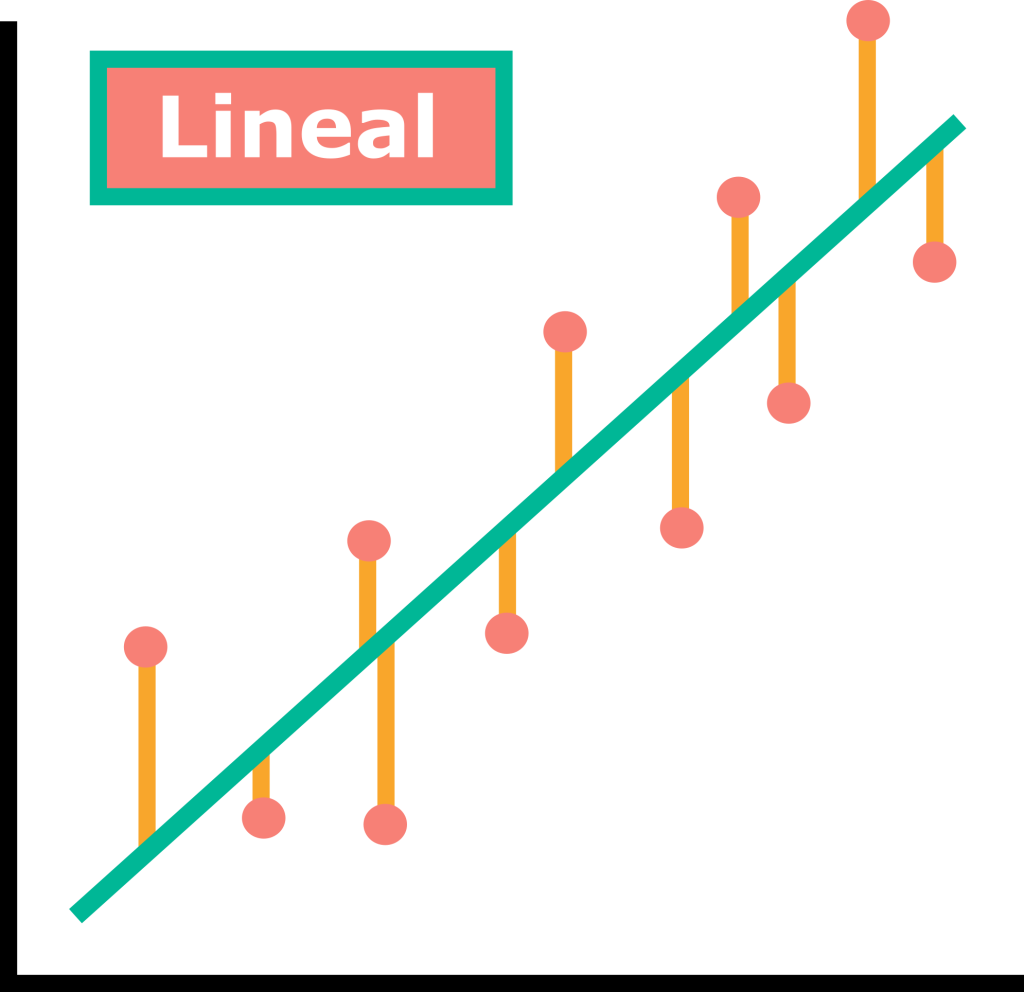
## **TAREA 1 (23-24) Regresión lineal con Python**



**Adrián Yared Armas de la Nuez**

**Índice**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

[**1. Actividad 2**](#_uw7mb35rettr)

[**1.1 Enunciado 2**](#_l3bc5ph592)

[**1.2 Desarrollo 2**](#_dzj4utiemd48)

[**1.2 Resultado 3**](#_laoitz6j3p79)

[**2. Actividad 3**](#_8m03s2xni85o)

[**2.1 Enunciado 3**](#_u42jlaqizsqm)

[**2.2 Desarrollo 4**](#_25a9ansltjw1)

[**2.3 Resultado 5**](#_vngpy9wd7hrv)

[**3. Actividad 5**](#_3n6uuej955ew)

[**3.1 Enunciado 5**](#_k1conmgws766)

[**3.2 Desarrollo 5**](#_weyin13rzbk8)

[**3.3 Resultado 7**](#_3hsf4m1dsbll)

**4**[**. Enlace del colab 7**](#_vuf1ms7y05ck)

## 

## **1. Actividad**

### **1.1 Enunciado**

Modifica el código usando los datos del archivo adjunto y muestra los puntos

junto con la recta de regresión usando el modelo de scikit-learn.

### **1.2 Desarrollo**

En esta primera actividad he utilizado librerías de python para realizar los cálculos.

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

# Crear el dataset

data = {

'restaurante': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],

'xi': [2, 6, 8, 8, 12, 16, 20, 20, 22, 26],

'yi': [58, 105, 88, 118, 117, 137, 157, 169, 149, 202]

}

# Convertir el dataset a un DataFrame

df = pd.DataFrame(data)

# Definir las variables X Y

X = df[['xi']]

Y = df['yi']

model = LinearRegression() # Crear el modelo

model.fit(X, Y) # Entrenar el modelo

Y\_pred = model.predict(X) # Hacer predicciones

# Dibujar los puntos de datos reales y la línea de regresión lineal

# puntos de datos reales (en color azul)

plt.scatter(X, Y, color='blue', label='Datos reales') # Los puntos representan X e Y

# línea de regresión (en color rojo)

plt.plot(X, Y\_pred, color='red', label='Línea de regresión') # La línea de regresión representa la predicción de Y en función de X

# Etiquetas para el eje X e Y

plt.xlabel('xi') # X,var independiente

plt.ylabel('yi') # Y, var dependiente

plt.title('Regresión Lineal - Restaurantes vs Ingresos') # Título del gráfico

plt.legend() # Mostrar la leyenda (puntos de datos reales y la línea de regresión)

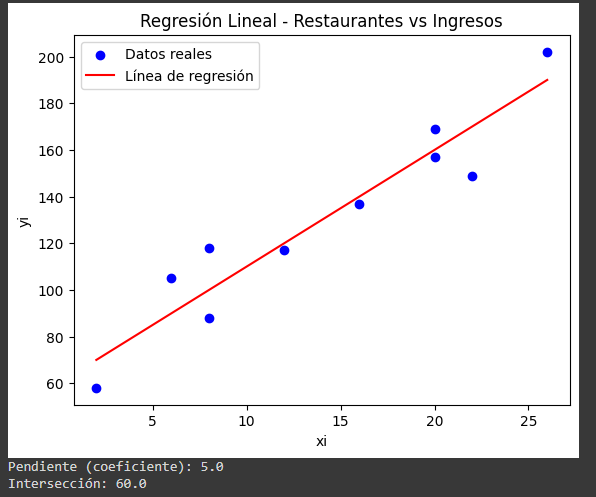
plt.show() # muestra el gráfico con los puntos y la línea de regresión

# Mostrar el coeficiente y la intersección de la recta

print(f'Pendiente (coeficiente): {model.coef\_[0]}')

print(f'Intersección: {model.intercept\_}')

### **1.2 Resultado**



## **2. Actividad**

### **2.1 Enunciado**

Calcula la recta de regresión usando las fórmulas y dibújala con matplotlib.

### **2.2 Desarrollo**

En esta segunda actividad he utilizado las fórmulas matemáticas para realizar el cálculo de la actividad.

import numpy as np

import matplotlib.pyplot as plt

# Definir los valores de xi y yi

xi = np.array([2, 6, 8, 8, 12, 16, 20, 20, 22, 26])

yi = np.array([58, 105, 88, 118, 117, 137, 157, 169, 149, 202])

# Calcular los valores necesarios para las fórmulas

# Calcular los valores necesarios para las fórmulas de regresión lineal

n = len(xi) # total puntos (xi, yi)

sum\_xi = np.sum(xi) # Suma de valores var independiente (xi)

sum\_yi = np.sum(yi) # Suma de valores var dependiente (yi)

sum\_xi\_yi = np.sum(xi \* yi) # Suma de cada (xi \* yi)

sum\_xi\_squared = np.sum(xi \*\* 2) # Suma (^2) de cada valor de xi

# Calcular la pendiente (m) con la form de regresión lineal

m = (n \* sum\_xi\_yi - sum\_xi \* sum\_yi) / (n \* sum\_xi\_squared - sum\_xi\*\*2)

# Calcular la intersección (b) en el eje y utilizando la fórmula

b = (sum\_yi - m \* sum\_xi) / n

# Imprimir los resultados

print(f'Pendiente (m): {m}')

print(f'Intersección (b): {b}')

# predicción valores yi

yi\_pred = m \* xi + b # Ecuación de la línea (y = m \* x + b)

# Graficar los puntos de datos originales y la línea de regresión

plt.scatter(xi, yi, color='blue', label='Datos reales') # Puntos de datos en color azul

plt.plot(xi, yi\_pred, color='red', label='Línea de regresión') # Línea de regresión en color rojo

# Etiquetas del gráfico

plt.xlabel('xi (Variable independiente)')

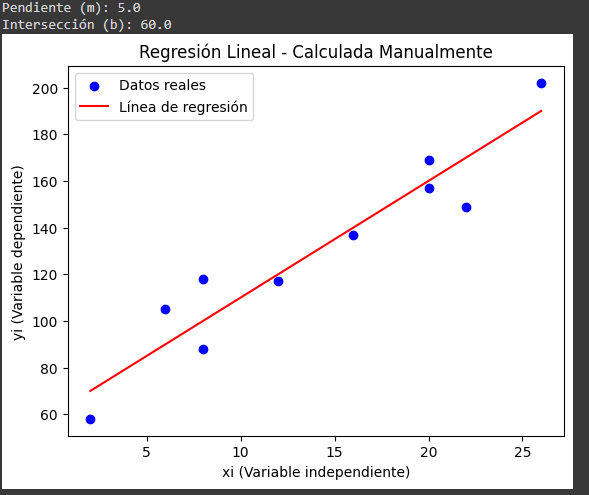
plt.ylabel('yi (Variable dependiente)')

plt.title('Regresión Lineal - Calculada Manualmente')

plt.legend() # Mostrar la leyenda (datos reales y línea de regresión)

plt.show() # Mostrar el gráfico

### **2.3 Resultado**



## **3. Actividad**

### **3.1 Enunciado**

Calcula los coeficientes de determinación r2 y r.

### **3.2 Desarrollo**

Para esta actividad la he utilizado la librería de cálculo de python.

import numpy as np

import matplotlib.pyplot as plt

# Definir los valores de xi y yi como arreglos de numpy

xi = np.array([2, 6, 8, 8, 12, 16, 20, 20, 22, 26]) # Valores independientes

yi = np.array([58, 105, 88, 118, 117, 137, 157, 169, 149, 202]) # Valores dependientes

# Calcular los valores para fórmulas regresión

n = len(xi) # N puntos

sum\_xi = np.sum(xi) # Suma de los xi

sum\_yi = np.sum(yi) # Suma de los yi

sum\_xi\_yi = np.sum(xi \* yi) # Sumas (xi \* yi)

sum\_xi\_squared = np.sum(xi \*\* 2) # Suma de los xi\*\*2

# Calcular la pendiente (m) y la intersección (b)

m = (n \* sum\_xi\_yi - sum\_xi \* sum\_yi) / (n \* sum\_xi\_squared - sum\_xi\*\*2) # Fórmula pendiente

b = (sum\_yi - m \* sum\_xi) / n # Fórmula intersección y

# predicciones yi (recta de regresión) con ecuación línea (y = m \* x + b)

yi\_pred = m \* xi + b

# coeficiente de determinación R^2 (evaluar el ajuste del modelo)

sst = np.sum((yi - np.mean(yi))\*\*2) # Suma cuadrados (variación total en yi)

ssr = np.sum((yi - yi\_pred)\*\*2) # Suma cuadrados de los residuos (variación no explicada)

# Coeficiente de determinación R^2, mide % de variabilidad de yi explicado por xi

r2 = 1 - (ssr / sst)

# Coeficiente de correlación R, que mide relación entre xi y yi

r = np.sqrt(r2)

# Mostrar los resultados

print(f'Coeficiente de determinación (R^2): {r2}')

print(f'Coeficiente de correlación (R): {r}')

### **3.3 Resultado**



## **4. Enlace del colab**

Dejo adjunto el [enlace del Colab](https://colab.research.google.com/drive/1B8jLop70ik9kxqS4c70klMMaYWDfzBoy?usp=sharing)