Taller 04 - Minimos Cuadrados

Nombre: Alexis Bautista
Fecha: 14 de diciembre de 2024
Link GitHub: https://github.com/alexis-bautista/TallerMinimosCuadrados-MN.git

Codigo proporcionado

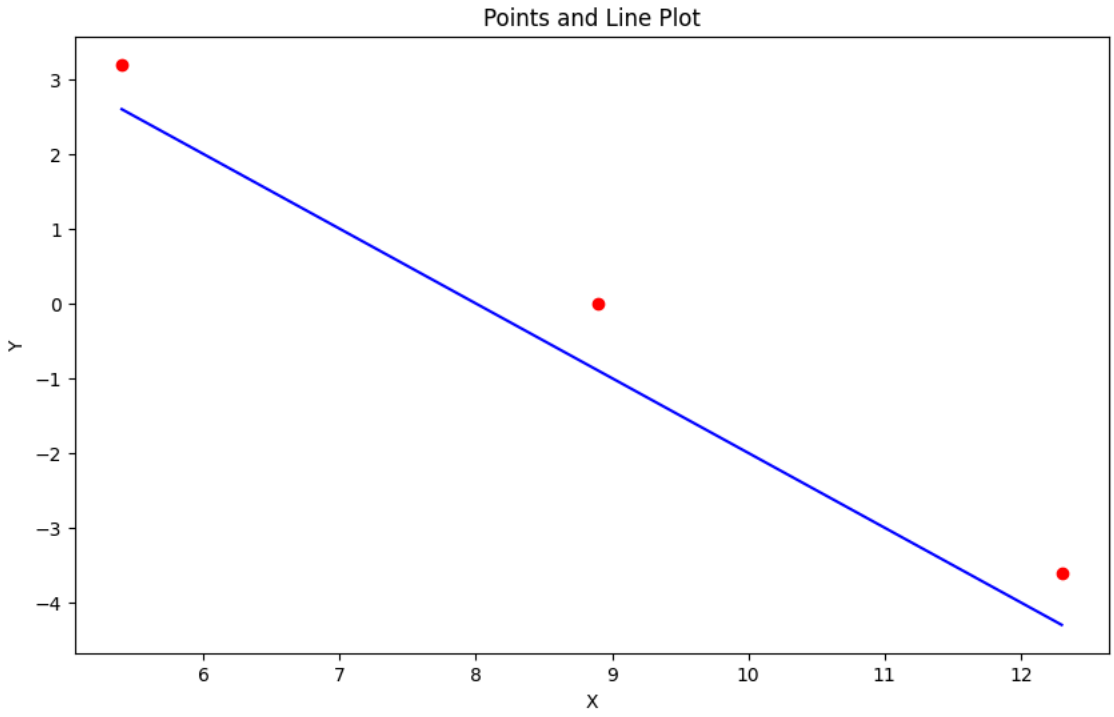
```
p1 = (5.4, 3.2)
p2_i = (9.5, 0.7)
p3 = (12.3, -3.6)
```

```
from ipywidgets import interact
import matplotlib.pyplot as plt

m = -1
b = 8
```

```
def update_plot(p2_x, p2_y):
    x_coords = [p1[0], p2_x, p3[0]]
    y_coords = [p1[1], p2_y, p3[1]]

    plt.figure(figsize=(10, 6))
    plt.scatter(x_coords, y_coords, color="red")

    x_line = [min(x_coords), max(x_coords)]
    y_line = [m * x + b for x in x_line]
    plt.plot(x_line, y_line, color="blue")

    plt.xlabel("X")
    plt.ylabel("Y")
    plt.title("Points and Line Plot")
    plt.show()
```

```
_ = interact(update_plot, p2_x=(5.5, 12.3, 0.1), p2_y=(-10.0, 10.0, 0.1))
```

p2_x 8.90
p2_y 0.00



Modificacion de codigo

Tenemos que el codigo proprcionado declara los valores de m y b, entonces en este caso agregamos regresion lineal para calcular automaticamente m y b se usa la libreria numpy con su funciones np.linalg.lstsq y numpy.vstack con esto logramos obtener una línea ajustada automáticamente, mejorando la representación visual.

Funcionamiento de las librerias numpy.vstack y numpy.linalg.lstsq
La función np.vstack se utiliza aquí para construir la matriz A requerida para calcular los parametros de la regresión lineal (m y b) mediante mínimos cuadrados.

np.linalg.lstsq resuelve el sistema de ecuaciones mediante mínimos cuadrados.

```
import numpy as np

def update_plot(p2_x, p2_y):
    x_coords = np.array([p1[0], p2_x, p3[0]])
    y_coords = np.array([p1[1], p2_y, p3[1]])

    # regresion lineal para calcular m y b
    A = np.vstack([x_coords, np.ones(len(x_coords))]).T
    '''np.ones Se usa para agregar una columna de unos a la matriz de entrada
    (A), lo que permite calcular el término constante (b) en la ecuación de la
    recta'''
    m, b = np.linalg.lstsq(A, y_coords, rcond=None)[0]

    plt.figure(figsize=(10, 6))
    plt.scatter(x_coords, y_coords, color="red")

    x_line = np.linspace(min(x_coords), max(x_coords), 100)
    y_line = m * x_line + b
    plt.plot(x_line, y_line, color="blue")

    plt.xlabel("X")
    plt.ylabel("Y")
    plt.title("Points and Line Plot with Linear Regression")
    plt.show()


_ = interact(update_plot, p2_x=(5.5, 12.3, 0.1), p2_y=(-10.0, 10.0, 0.1))
```

p2_x 8.90

p2_y 0.00