Corrección

Boris Garcés

Tabla de Contenidos

Repositorio

https://github.com/Boris-epn/prueba-correccion

Mínimos cuadrados

Prueba 02

Interpole los siguientes conjuntos de datos con la función correspondiente.

La ecuación de la línea es:

$ y(x) = a_1 x + a_0 $

Al realizar el proceso de mínimos cuadrados queda el siguiente sistema de ecuaciones:

$(\sum_i (y_i - a_1 x_i - a_0), \sum_i (y_i - a_1 x_i - a_0) x_i) = 0$

```
def der_parcial_2(xs, ys):
    return [sum(x**4 for x in xs), sum(x**3 for x in xs), sum(x**2 for x in xs), sum(y * x**

def der_parcial_1(xs, ys):
    return [sum(x**3 for x in xs), sum(x**2 for x in xs), sum(x for x in xs), sum(y * x for 

def der_parcial_0(xs, ys):
    return [sum(x**2 for x in xs), sum(x for x in xs), len(xs), sum(ys)]
```

Conjunto de datos de ejemplo

```
from src import ajustar_min_cuadrados  # no modificar esta función

pars = ajustar_min_cuadrados(xs, ys, gradiente=[der_parcial_0, der_parcial_1])
```

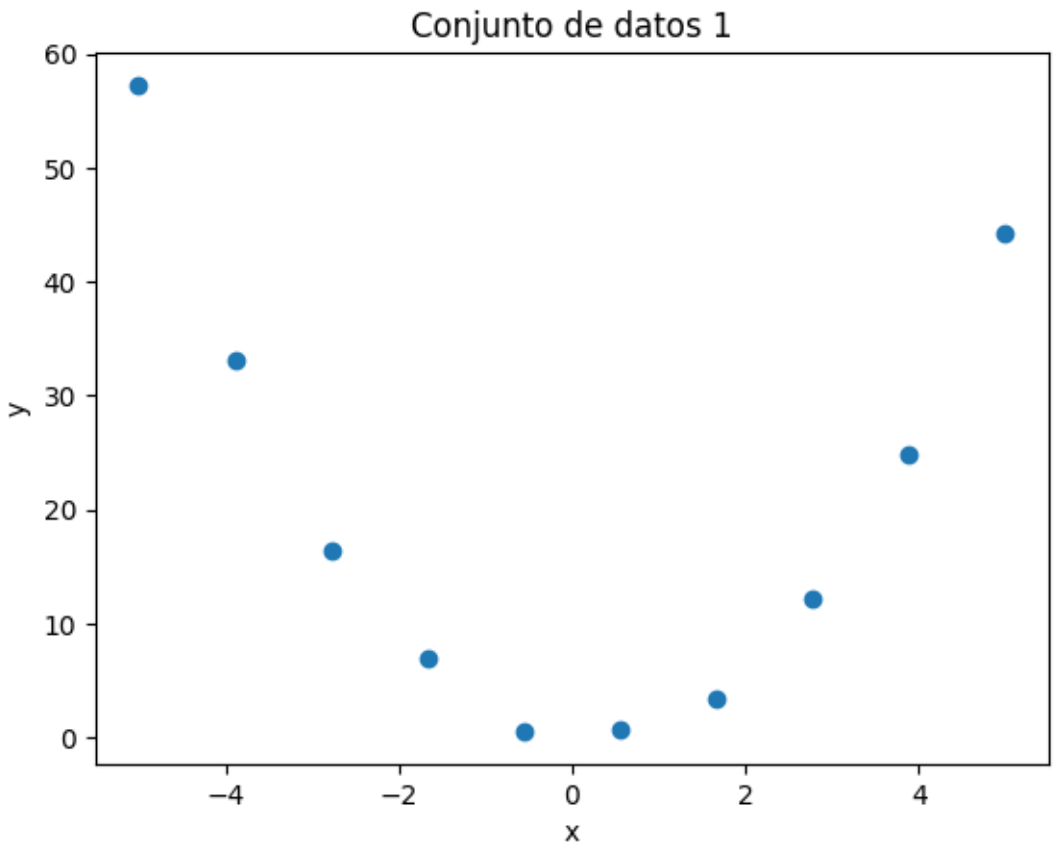
```
[01-23 12:05:25][INFO] 2025-01-23 12:05:25.820490
[01-23 12:05:25][INFO] 2025-01-23 12:05:25.829680
[01-23 12:05:25][INFO] Se ajustarán 2 parámetros.
[01-23 12:05:25][INFO]
[[101.8525926    0.         209.87476711]
 [  0.          10.         -11.7356    ]]
```

Conjunto de datos 1

```
xs1 = [
    -5.0000,
    -3.8889,
    -2.7778,
    -1.6667,
    -0.5556,
    0.5556,
    1.6667,
    2.7778,
    3.8889,
    5.0000,
]
```

```
ys1 = [
    57.2441,
    33.0303,
    16.4817,
    7.0299,
    0.5498,
    0.7117,
    3.4185,
    12.1767,
    24.9167,
    44.2495,
]
```

```
plt.scatter(xs1, ys1)

plt.xlabel("x")
plt.ylabel("y")
plt.title("Conjunto de datos 1")
plt.show()
```

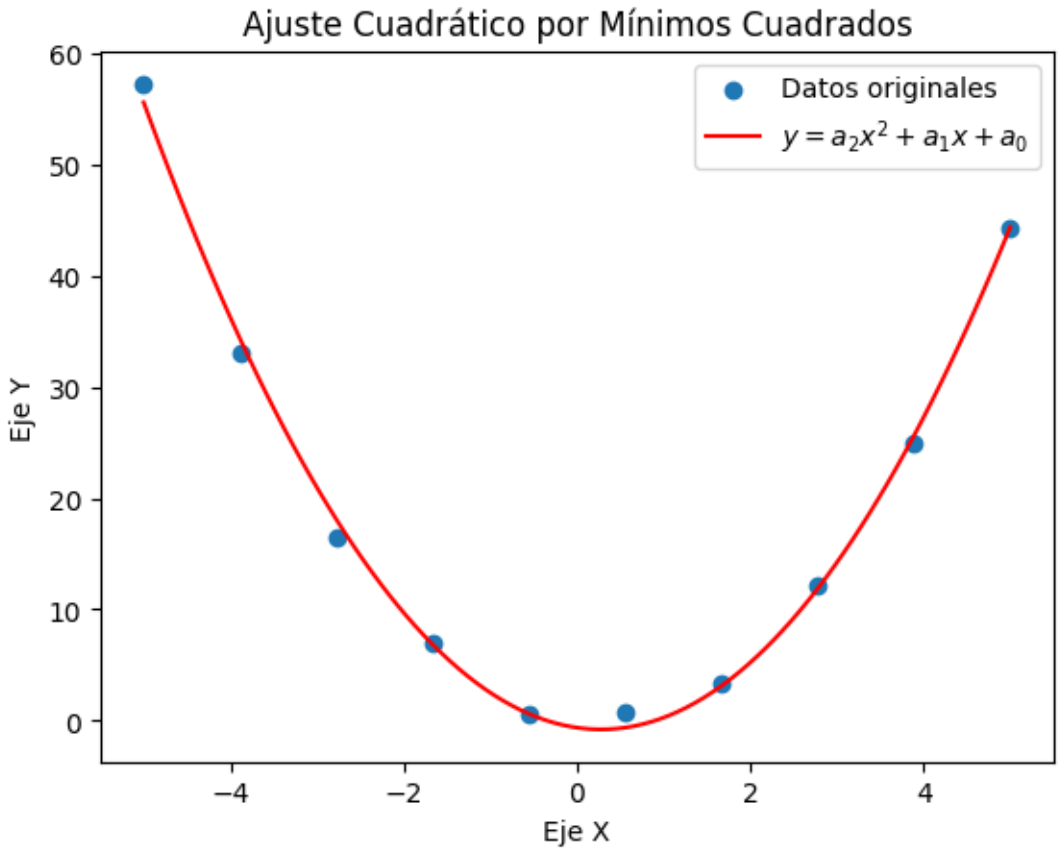



```
def der_parcial_2(xs, ys):
    return [sum(x**4 for x in xs), sum(x**3 for x in xs), sum(x**2 for x in xs), sum(y * x**

def der_parcial_1(xs, ys):
    return [sum(x**3 for x in xs), sum(x**2 for x in xs), sum(x for x in xs), sum(y * x for

def der_parcial_0(xs, ys):
    return [sum(x**2 for x in xs), sum(x for x in xs), len(xs), sum(ys)]
```

```
a2, a1, a0 = ajustar_min_cuadrados(xs1, ys1, gradiente=[der_parcial_2, der_parcial_1, der_pa
x_rango = np.linspace(-5, 5, 100)
y_rango = [a2 * x**2 + a1 * x + a0 for x in x_rango]
plt.scatter(xs1, ys1, label="Datos originales")
plt.plot(x_rango, y_rango, color="red", label=r"$y = a_2 x^2 + a_1 x + a_0$")
plt.xlabel("Eje X")
plt.ylabel("Eje Y")
plt.title("Ajuste Cuadrático por Mínimos Cuadrados")
```

```
plt.legend()
plt.show()
```

```
[01-23 12:24:08][INFO] Se ajustarán 3 parámetros.
[01-23 12:24:08][INFO]
[[ 1.01852593e+02  0.00000000e+00  1.00000000e+01  1.99808900e+02]
 [ 0.00000000e+00  1.01852593e+02  0.00000000e+00 -1.14413577e+02]
 [-2.27373675e-13  0.00000000e+00 -7.90113041e+01  5.04294087e+01]]
[01-23 12:24:08][INFO]
[[ 1.01852593e+02  0.00000000e+00  1.00000000e+01  1.99808900e+02]
 [ 0.00000000e+00  1.01852593e+02  0.00000000e+00 -1.14413577e+02]
 [-2.27373675e-13  0.00000000e+00 -7.90113041e+01  5.04294087e+01]]
```



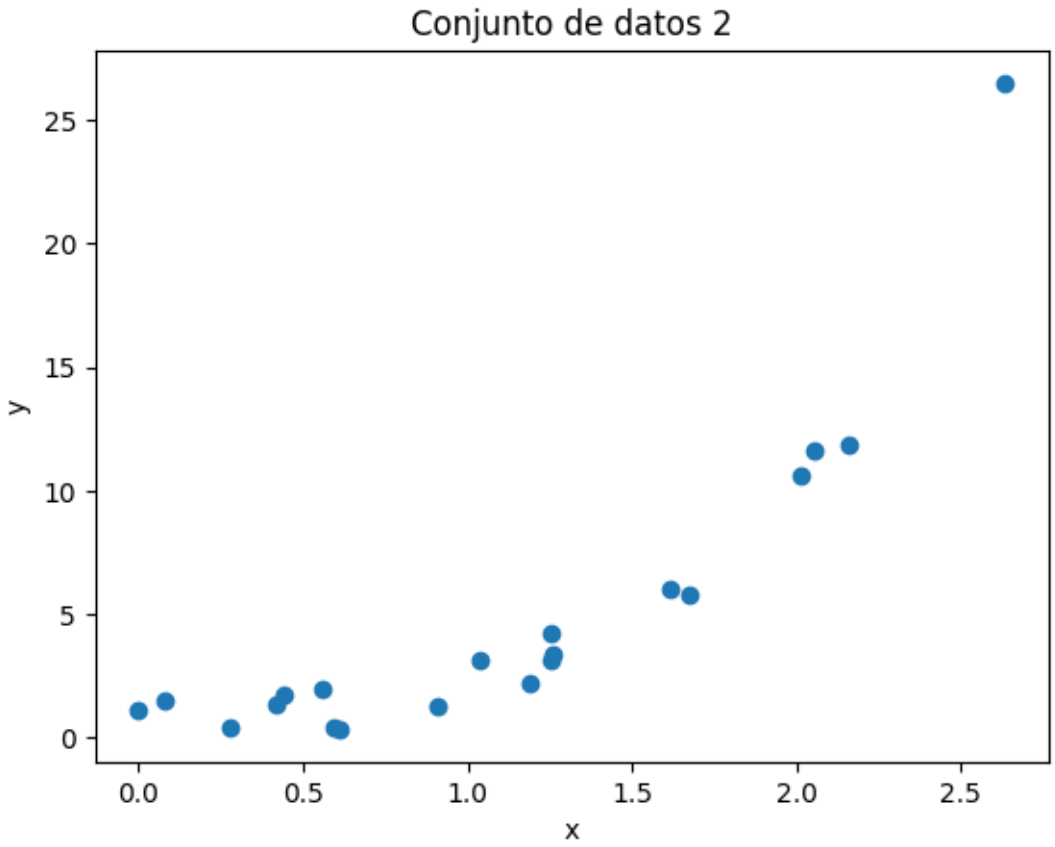
Interpole el conjunto de datos 1 usando la función cuadrática.

Conjunto de datos 2

```
xs2 = [
    0.0003,
    0.0822,
    0.2770,
    0.4212,
    0.4403,
    0.5588,
    0.5943,
    0.6134,
    0.9070,
    1.0367,
    1.1903,
    1.2511,
    1.2519,
    1.2576,
    1.6165,
    1.6761,
    2.0114,
    2.0557,
    2.1610,
    2.6344,
]
ys2 = [
    1.1017,
    1.5021,
    0.3844,
    1.3251,
    1.7206,
    1.9453,
    0.3894,
    0.3328,
    1.2887,
    3.1239,
    2.1778,
    3.1078,
    4.1856,
    3.3640,
    6.0330,
    5.8088,
    10.5890,
```

```
    11.5865,
    11.8221,
    26.5077,
]
```

```
plt.scatter(xs2, ys2)
plt.xlabel("x")
plt.ylabel("y")
plt.title("Conjunto de datos 2")
plt.show()
```

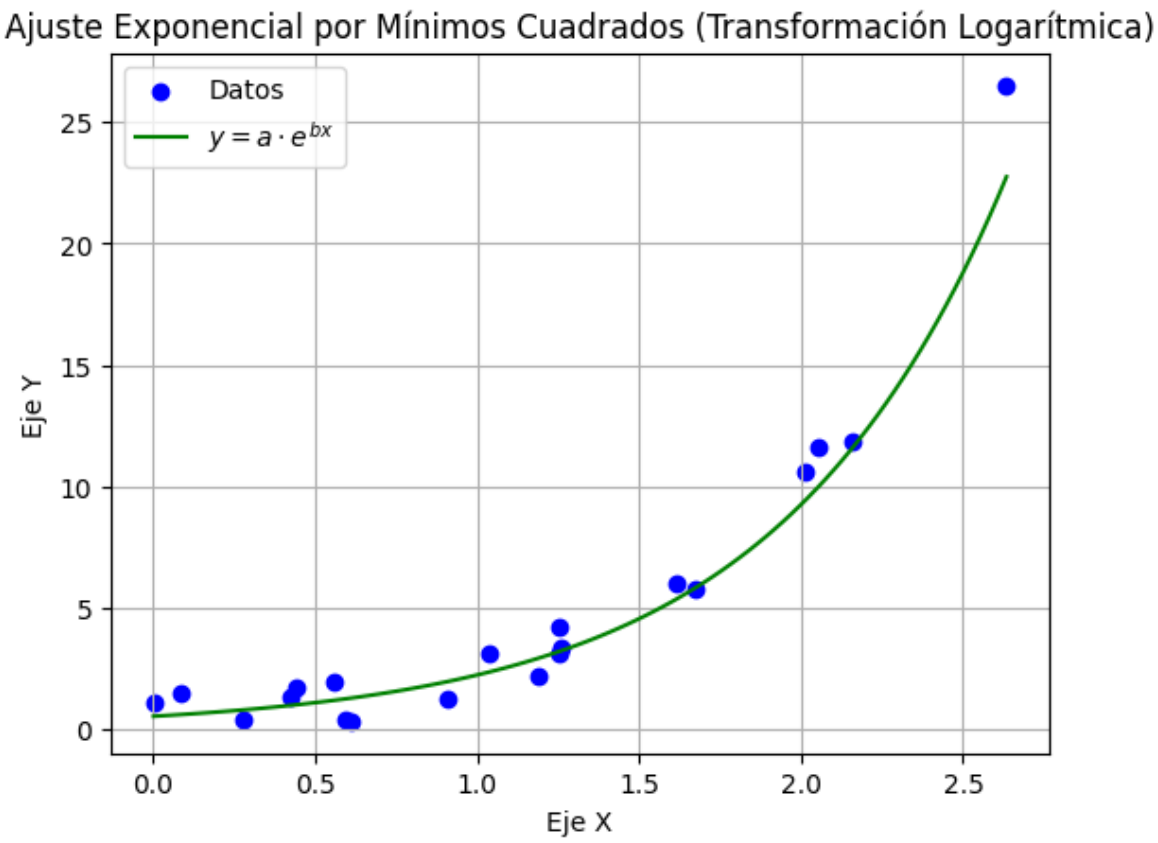


Interpole el conjunto de datos 2 usando la función exponencial.

```
def calcular_derivada_parcial_a(xs, ys):
    log_ys = np.log(ys)
    return [
        len(xs),
```

```
        sum(xs),
        sum(log_ys),
    ]

def calcular_derivada_parcial_b(xs, ys):
    log_ys = np.log(ys)
    return [
        sum(xs),
        sum(x**2 for x in xs),
        sum(x * y for x, y in zip(xs, log_ys)),
    ]
A_primado, b = ajustar_min_cuadrados(xs2, ys2, gradiente=[calcular_derivada_parcial_a, calcul
a = np.exp(A_primado)
x_ajuste = np.linspace(min(xs2), max(xs2), 100)
y_ajuste = [a * np.exp(b * x) for x in x_ajuste]
plt.scatter(xs2, ys2, label="Datos", color="blue")
plt.plot(x_ajuste, y_ajuste, color="green", label=r"$y = a \cdot e^{b x}$")
plt.xlabel("Eje X")
plt.ylabel("Eje Y")
plt.title("Ajuste Exponencial por Mínimos Cuadrados (Transformación Logarítmica)")
plt.legend()
plt.grid()
plt.show()
```

```
[01-23 12:25:51][INFO] Se ajustarán 2 parámetros.
[01-23 12:25:51][INFO]
[[20.         22.0372     19.05727035]
 [ 0.         10.54683259 14.94655314]]
```


Ajuste Exponencial por Mínimos Cuadrados (Transformación Logarítmica)
Datos
$y = a \cdot e^{bx}$
Eje Y
Eje X
0
5
10
15
20
25
0.0
0.5
1.0
1.5
2.0
2.5