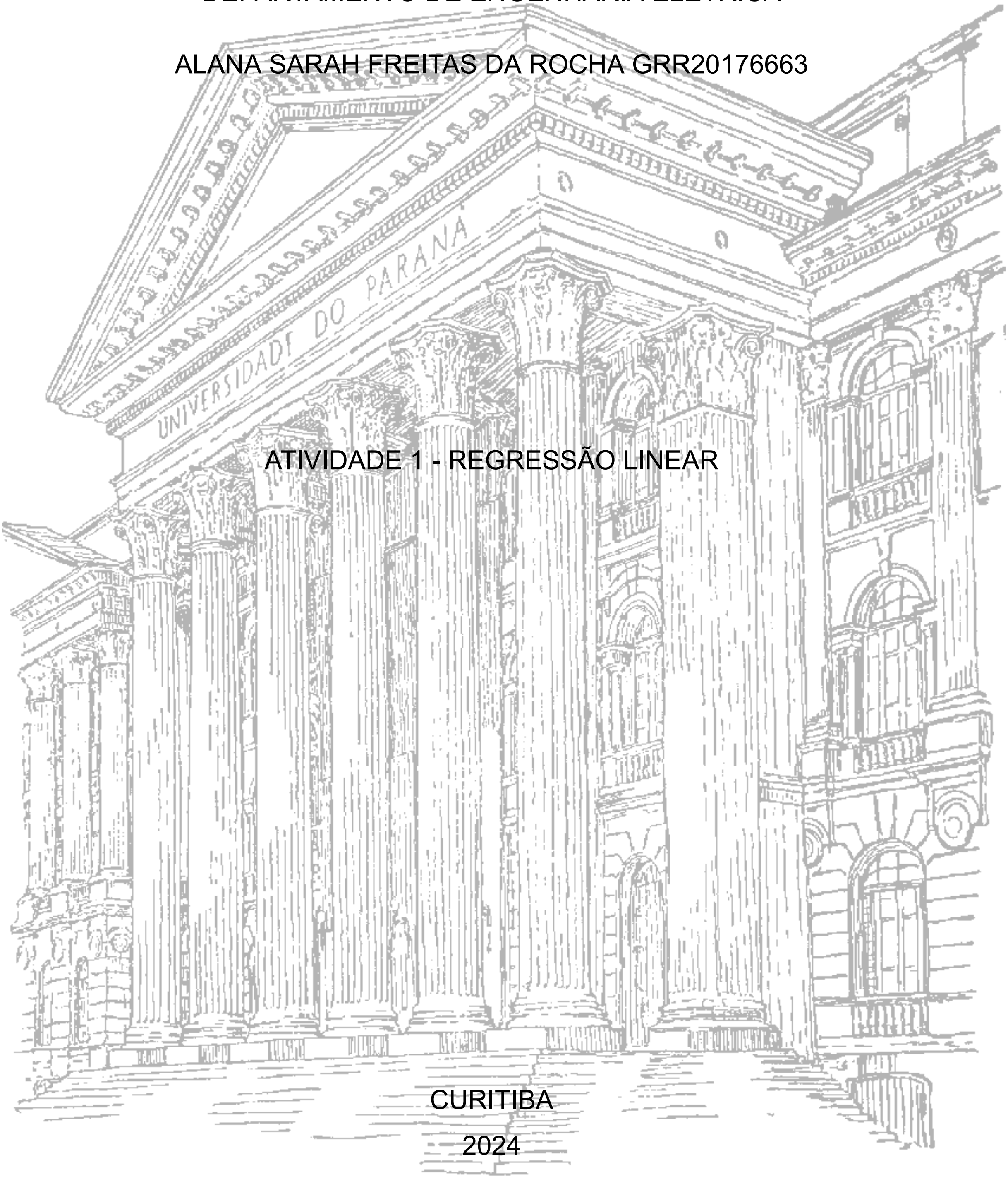
UNIVERSIDADE FEDERAL DO PARANÁ
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

ALANA SARAH FREITAS DA ROCHA GRR20176663



ATIVIDADE 1 - REGRESSÃO LINEAR

CURITIBA
2024

SUMÁRIO

1.INTRODUÇÃO

Nesta atividade vamos aplicar a regressão linear simples. O objetivo é adequar uma reta aos dados fornecidos, reduzindo ao máximo o erro entre os valores previstos e os observados. Para isso, vamos considerar duas abordagens: uma resolução manual utilizando derivadas e sistemas de equações a partir do método dos mínimos quadrados, e uma resolução matricial utilizando Python, com base em um enunciado fornecido.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 REGRESSÃO LINEAR

A regressão linear simples modela a relação entre duas variáveis: uma variável independente X e uma variável dependente Y. O objetivo é encontrar uma função linear da forma:

$$Y = a \cdot X + b$$

Onde:

- a é o coeficiente angular ou inclinação da reta,
- b é o ponto em que a reta intercepta o eixo.

A regressão linear busca ajustar os coeficientes a e b para que a curva se ajuste de forma mais eficaz aos dados coletados, reduzindo o erro entre os valores previstos e os reais de Y.

2.2 MÉTODO DOS MÍNIMOS QUADRADOS

O método dos mínimos quadrados é utilizado para encontrar os coeficientes a e b que minimizam o Erro Quadrático Médio (MSE). A função de erro quadrático médio é a seguinte:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} \left(Y_i - (a \cdot X_i + b)\right)^2$$

Onde:

- Y_i são os valores observados da variável dependente,
- $\widehat{Y}_i = a \cdot X_i + b$ são os valores previstos pela reta,
- n é o número de observações.

2.3 ENUNCIADO DA ATIVIDADE

Seja o conjunto de 5 medições (xi, yi), onde i=1, 2, ..., 5:

(x1, y1)= (0,0.2)

(x2, y2)= (0.1,0.3)

(x3, y3)= (0.2,0.45)

(x4, y4)= (0.3,0.7)
(x5, y5)= (0.4,0.8)

Deseja-se obter a melhor reta que relaciona (xi, yi). Essa reta terá a seguinte equação yi=a.xi+b, onde a e b são constantes ainda a serem determinadas. Obtenha os valores otimizados para os coeficientes a e b, usando mínimos quadrados de duas formas distintas.

2.4 RESOLUÇÃO MANUAL

Seja O conjunto de 5 medições fornecidos, deseja-se obter a melhor reta

$(x_1, y_1) = (0, 0.2)$

$(x_2, y_2) = (0.1, 0.3)$

$(x_3, y_3) = (0.2, 0.45)$

$(x_4, y_4) = (0.3, 0.7)$

$(x_5, y_5) = (0.4, 0.8)$

sendo : $y_i = a x_i + b$

Erro quadrático médio (MSE) :

$$MSE(a,b) = \frac{1}{n} \sum_{i=1}^{n} (y_i - (a x_i + b))^2$$

1-) Derivar MSE em relação a a e b

1.1) Derivar em relação a a:

$$\frac{1}{n} \sum_{i=1}^{n} (y_i - (a x_i + b))^2$$

constante, podemos ignorar

aplicar regra da cadeia pois a função $(y_i - (a x_i + b))^2$ é uma função composta :

$$\frac{d}{da} (y_i - (a x_i + b))^2 = 2(y_i - (a x_i + b)) \frac{d}{da} (y_i - (a x_i + b))$$

$$\frac{d}{da} (y_i - (a x_i + b)) = -x_i$$

assim : $$\frac{d}{da} S(a,b) = -2 \sum_{i=1}^{n} x_i (y_i - (a x_i + b))$$

1.2) Derivar em relação a b:

$$\frac{1}{n}\sum_{i=1}^{n}(yi-(axi+b))^2$$

↳ como no exemplo anterior podemos ignorar por ser uma constante

$$\frac{d}{db}(yi-(axi+b))^2 = 2(yi-(axi+b))\frac{d}{db}(yi-(axi+b))$$

$$\frac{d}{db}(\cancel{yi}-(\cancel{a}xi+b)) = -1$$

$$\quad 0 \quad 0$$

assim temos:

$$\frac{d}{db}s(a,b) = -2\sum_{i=1}^{n}(yi-(axi+b))$$

1.3) Igualamos as duas derivadas a 0

$\frac{d}{da}$ igual a 0:

$$0 = -2\sum_{i=1}^{n} xi(yi-(axi+b))$$

$$\boxed{\sum_{i=1}^{n} xi\,yi = a\sum_{i=1}^{n} xi^2 + b\sum_{i=1}^{n} xi}$$

$\frac{d}{db}$ igual a 0:

$$0 = -2\sum_{i=1}^{n}(yi-(axi+b))$$

$$0 = -2\sum_{i=1}^{n} yi + 2a\sum_{i=1}^{n} xi + 2b\sum_{i=1}^{n}$$

$$\boxed{\sum_{i=1}^{n} yi = a\sum_{i=1}^{n} xi + nb}$$

Assim temos um sistema de equações

$$1. \quad \sum_{i=1}^{n} x_i y_i = a \sum_{i=1}^{n} x_i^2 + b \sum_{i=1}^{n} x_i$$

$$2. \quad \sum_{i=1}^{n} y_i = a \sum_{i=1}^{} x_i + nb$$

2. Podemos calcular apartir dos valores fornecidos:

$n = 5$

$$\sum_{i=1}^{n} x_i = 0 + 0.1 + 0.2 + 0.3 + 0.4 = 1$$

$$\sum_{i=1}^{n} y_i = 0.2 + 0.3 + 0.45 + 0.7 + 0.8 = 2.45$$

$$\sum_{i=1}^{n} x_i^2 = 0 + 0.01 + 0.04 + 0.09 + 0.16 = 0.3$$

$$\sum_{i=1}^{n} x_i y_i = 0 + 0.03 + 0.9 + 0.21 + 0.32 = 0.65$$

Assim podemos substituir

$$0.65 = a\,0.3 + b$$

$$2.45 = a + 5.b$$

Resolvendo o sistema:

$$a = 2.45 - 5b$$

$$0.65 = 0.3(2.45 - 5b) + b$$

$$0.65 = 0.735 - 1{,}5b + b$$

$$-0.085 = -0{,}5b$$

$$b = 0.17$$

então

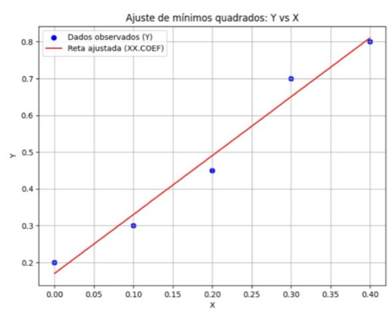
$$2.45 = a + 0{,}85$$

$$a = 1{,}6$$

assim podemos escrever a equação da reta ajustada como:

$$y_i = 1.6\,a + 0.17$$

Então podemos desenhar o gráfico:



3.RESOLUÇÃO UTILIZANDO NOTAÇÃO MATRICIAL

Podemos resolver o mesmo problema de maneira mais eficiente usando a notação matricial cuja equação é:

$$Y = X.\beta$$

Onde:

- Y é o vetor de saídas observadas,
- X é a matriz de regressão, que contém as variáveis $x_1, x_2, x_3, \ldots, x_n$ e uma coluna de 1's para incluir o intercepto *b*,
- β é o vetor de coeficientes ajustáveis

Para isso organizamos os valores observados da seguinte forma:

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \qquad X = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \qquad \beta = \begin{bmatrix} a \\ b \end{bmatrix}$$

A solução dos mínimos quadrados para β é obtida usando a seguinte fórmula que nos dá os coeficientes a e b:

$$\beta = (X^T X)^{-1} X^T Y$$

3.1 RESOLUÇÃO UTILIZANDO BIBLIOTECA NUMPY

O NumPy disponibiliza uma função eficiente para resolver sistemas de equações lineares através dos mínimos quadrados: linalg.lstsq. Esta função resolve o sistema $Y = X.\beta$ e retorna os coeficientes que minimizam o erro quadrático médio.

Aplicação do código:

```python
#Utilizando numpy
import numpy as np
import matplotlib.pyplot as plt

# Passo 1: Definindo os vetores e a matriz de regressão conforme os valores do enunciado
Y = np.array([0.20, 0.30, 0.45, 0.70, 0.80])
print(Y)
X = np.array([0.00, 0.10, 0.20, 0.30, 0.40])
print(X)
# 1c. Matriz de regressão XX (5x2)
XX = np.column_stack((X, np.ones(X.shape[0])))
print(XX)
```

[34] 0.0s Python

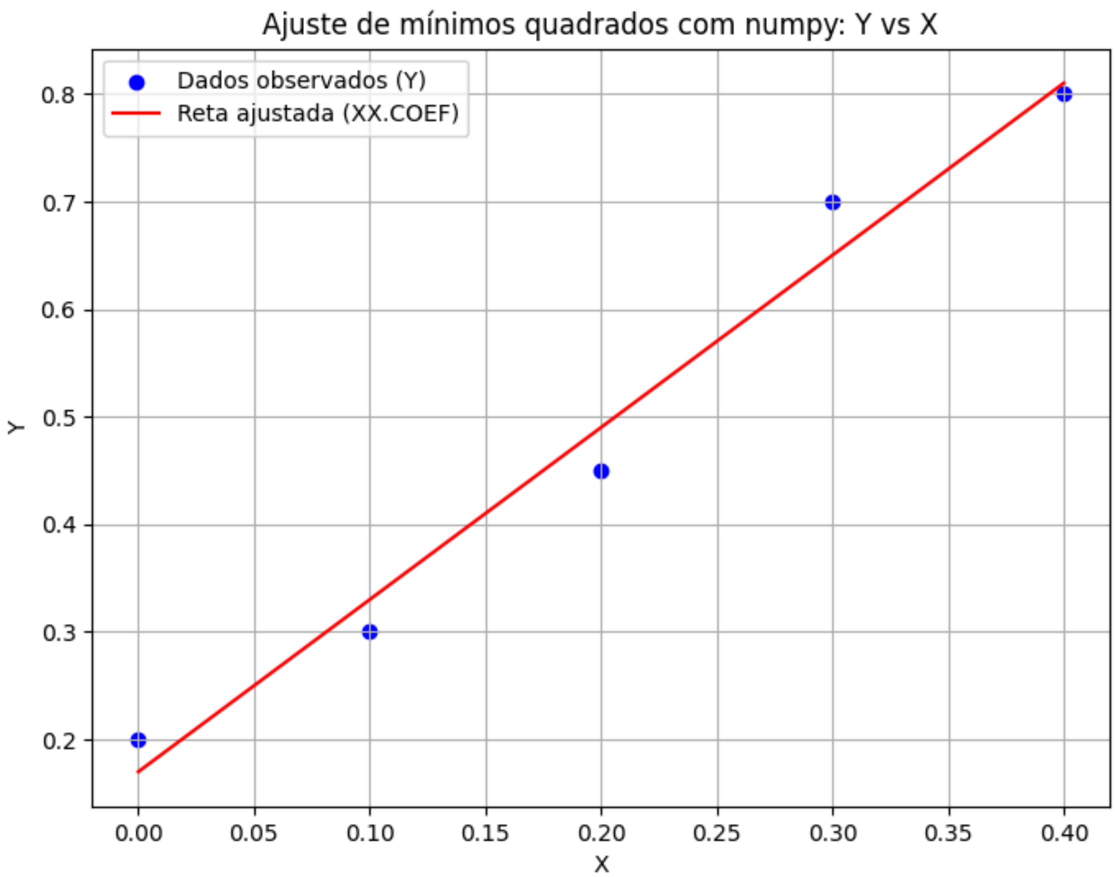
```
[0.2  0.3  0.45 0.7  0.8 ]
[0.  0.1 0.2 0.3 0.4]
[[0.  1. ]
 [0.1 1. ]
 [0.2 1. ]
 [0.3 1. ]
 [0.4 1. ]]
```

```python
# Passo 2: Resolvendo o sistema Y = XX.COEF usando o método dos mínimos quadrados
# Usamos np.linalg.lstsq para resolver a funcao Y=X·β e retornar os coeficientes que minimizam o erro quadrático médio.
COEF, residuals, rank, s = np.linalg.lstsq(XX, Y,rcond=None)
# Coeficientes a e b
a, b = COEF
print(f"\nCoeficiente a (inclinação): {a:.2f}")
print(f"Coeficiente b (interceptor): {b:.2f}")

Y_est = XX @ COEF  # Multiplicação matricial para obter os valores estimados
print(Y_est)
```
[37] ✓ 0.0s Python

```
Coeficiente a (inclinação): 1.60
Coeficiente b (interceptor): 0.17
[0.17 0.33 0.49 0.65 0.81]
```

```python
# Passo 4: Visualização
plt.figure(figsize=(8, 6))
plt.scatter(X, Y, color='blue', label='Dados observados (Y)')
plt.plot(X, Y_est, color='red', label='Reta ajustada (XX.COEF)')
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Ajuste de mínimos quadrados: Y vs X')
plt.legend()
plt.grid(True)
plt.show()
```
[38] ✓ 0.0s Python

Gráfico gerado:



3.2 RESOLUÇÃO UTILIZANDO BIBLIOTECA SCIKIT-LEARN

O Scikit-learn disponibiliza a classe LinearRegression, que simplifica o ajuste de modelos de regressão linear de maneira eficaz e possibilita sua aplicação em questões de aprendizado de máquina.

Código:

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# Passo 1: Definir os dados de entrada
X = np.array([0, 0.1, 0.2, 0.3, 0.4]).reshape(-1, 1)  # Redimensionar matriz
print(X)
Y = np.array([0.2, 0.3, 0.45, 0.7, 0.8])
print(Y)
```

[43] ✓ 0.0s Python

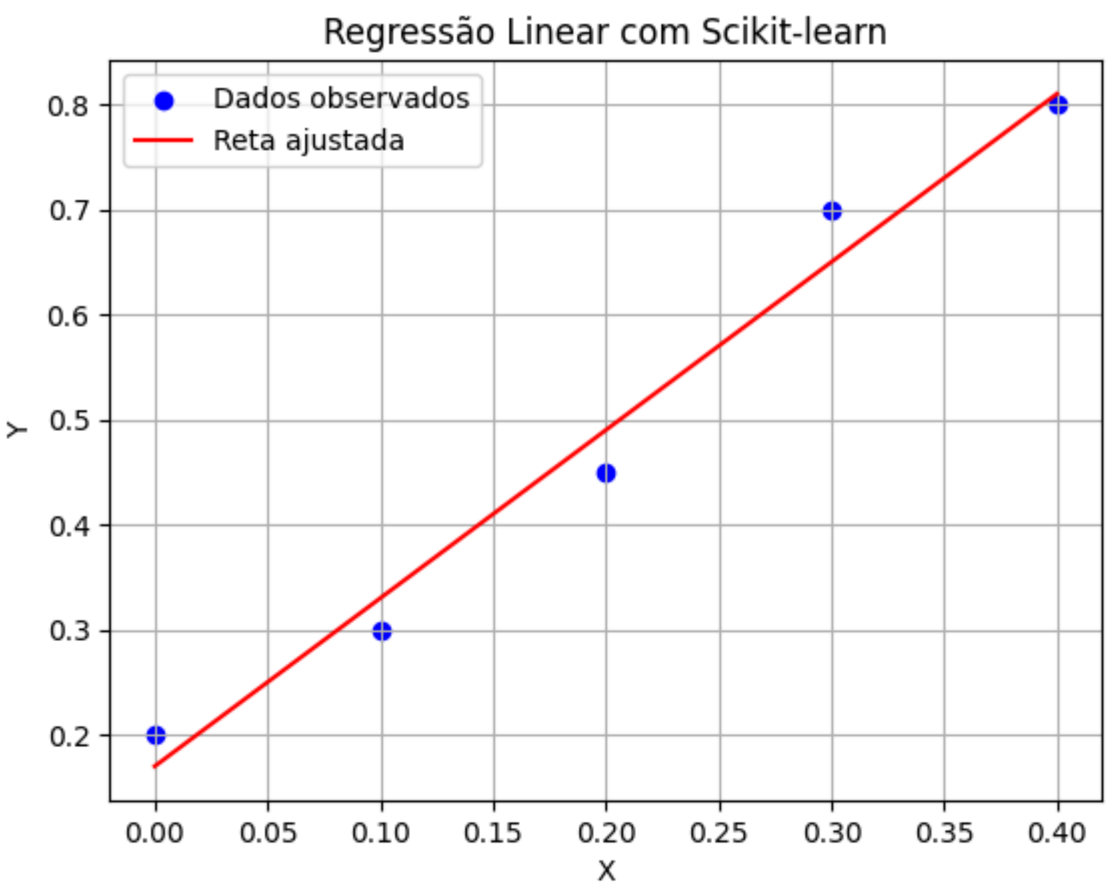
```
[[0. ]
 [0.1]
 [0.2]
 [0.3]
 [0.4]]
[0.2  0.3  0.45 0.7  0.8 ]
```

```python
# Passo 2: Criar e ajustar o modelo de regressão linear
model = LinearRegression()
model.fit(X, Y)

# Passo 3: Obter os coeficientes
a = model.coef_[0]  # Inclinação
b = model.intercept_  # Intercepto
print(f"Coeficiente a (inclinação): {a}")
print(f"Coeficiente b (intercepto): {b}")
# Passo 4: Fazer previsões com base nos valores de X. função aplica a equação da reta y =a·x+b para cada valor de x retornando os valores estimados y
Y_est = model.predict(X)
print(Y_est)
```

[45] ✓ 0.0s Python

```
Coeficiente a (inclinação): 1.6
Coeficiente b (intercepto): 0.16999999999999998
[0.17 0.33 0.49 0.65 0.81]
```

Gráfico gerado:



3.3 RESOLUÇÃO UTILIZANDO BIBLIOTECA SCIPY

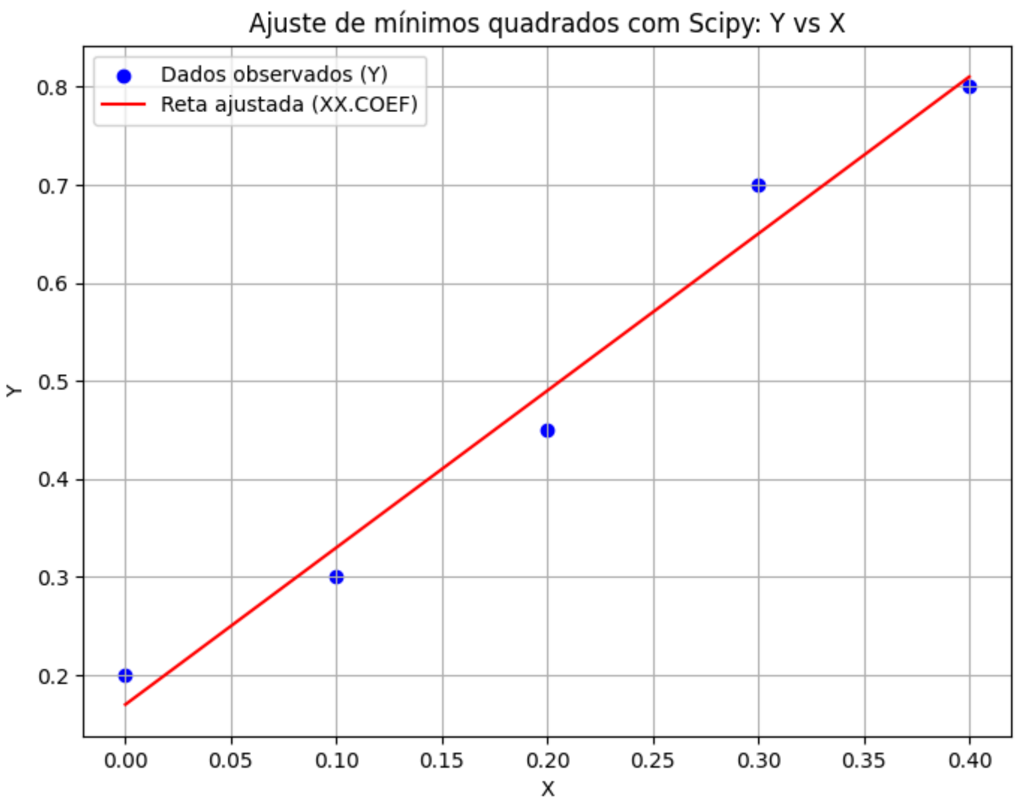
Para efeito de comparações, vamos também utilizar a biblioteca SciPy que disponibiliza a função scipy.linalg.lstsq, que se assemelha bastante à função np.linalg.lstsq do NumPy, contudo, possui mais recursos específicos para cálculos de álgebra linear.

Código:

```python
#Utilizando Scipy
import numpy as np
from scipy.linalg import lstsq

# Definindo os dados de entrada
X = np.array([0, 0.1, 0.2, 0.3, 0.4])
Y = np.array([0.2, 0.3, 0.45, 0.7, 0.8])

# Criando a matriz de regressão XX com uma coluna de 1s para o termo constante
XX = np.column_stack((X, np.ones(X.shape[0])))

# Resolver o sistema linear usando mínimos quadrados com scipy.linalg.lstsq
COEF, residuals, rank, s = lstsq(XX, Y)

# Extraindo os coeficientes a e b
a, b = COEF
print(f"Inclinação (a): {a:.3f}, Intercepto (b): {b:.3f}")
```

```
[32] 0.0s                                                    Python
... Inclinação (a): 1.600, Intercepto (b): 0.170
```

Gráfico gerado:



4 CONCLUSÃO

Nesta atividade, discutimos a questão de estabelecer os coeficientes a e b de uma reta que melhor se ajusta a um conjunto de dados por meio da regressão linear. O método dos mínimos quadrados é aplicado para reduzir a soma dos erros quadráticos, podemos realizar isso por meio de resolução manual ou por meio de código utilizando bibliotecas do Python.

No processo manual, utilizamos a técnica dos mínimos quadrados para derivar a função do erro em relação aos coeficientes a e b, gerando um conjunto de equações que foi solucionado para determinar os valores ótimos. Este procedimento nos possibilitou entender o princípio matemático por trás da regressão linear e destacou a relevância de reduzir o erro quadrático para determinar a reta de ajuste mais adequada.

Em seguida, resolvemos o problema utilizando a notação matricial comparando o resultado de três diferentes bibliotecas em Python: NumPy, SciPy, e Scikit-learn. Cada uma das bibliotecas oferece ferramentas para resolver o problema de maneira mais eficaz:

- NumPy: Permite a manipulação eficiente de arrays e matrizes e simplificou a resolução matricial do problema, onde calculamos os coeficientes diretamente a partir da equação matricial β.
- SciPy: Com foco maior em álgebra linear e resolução de equações, utilizamos a função scipy.linalg.lstsq para resolver o problema de mínimos quadrados de forma direta e simples.
- Scikit-learn: Mais utilizada para machine learning, a resolução com a função LinearRegression provou que a aplicação do modelo de regressão pode ser feita de forma mais abstrata e automatizada, ideal para casos em que se deseja construir e treinar modelos de forma rápida.

Quando analisamos as três estratégias, observamos que todas resultam nos mesmos coeficientes a e b, o que confirma a consistência entre os métodos manuais e computacionais.

5.REFERÊNCIAS

"Método dos Mínimos Quadrados" - UEL. Disponivel em <https://www.uel.br/projetos/matessencial/superior/alinear/mmq.html#sec01>

Algebra Linear - UFRGS. Disponivel em <https://www.ufrgs.br/reamat/AlgebraLinear/livro/s14-mx00e9todo_dos_mx00ednimos_quadrados.html>

Referência Numpy. Disponivel em <https://numpy.org/doc/stable/reference/generated/numpy.linalg.lstsq.html>

Referência Scikit-learn. Disponivel em <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html>

Referência Scipy. Disponivel em <https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.lstsq.html>