

Inteligência Artificial

2024/2



Profa. Dra. Juliana Félix

jufelix16@gmail.com



Regressão

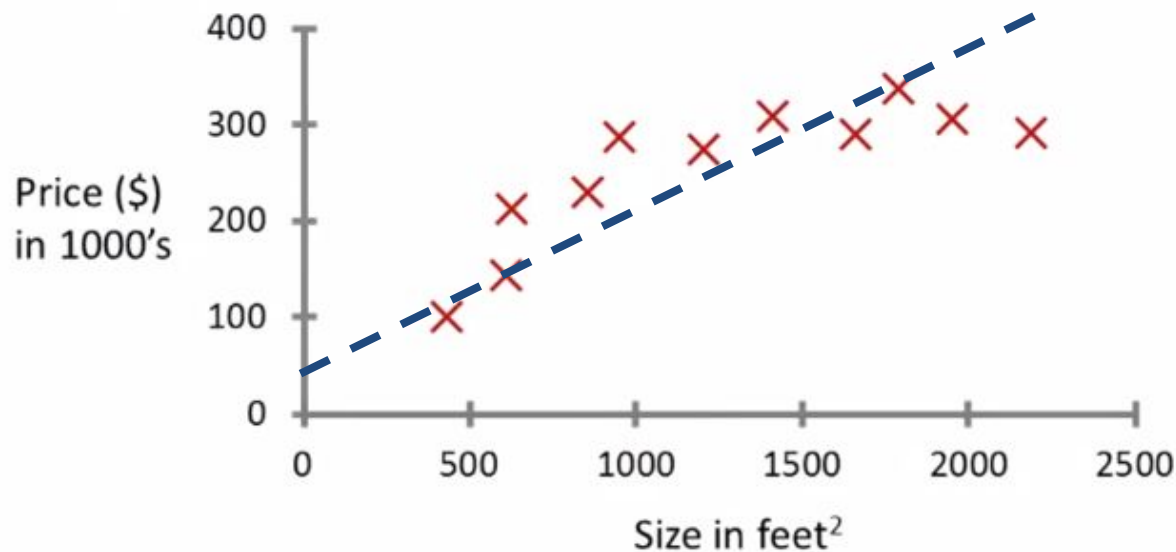
Regressão

A **regressão linear** é uma técnica de análise de dados que permite prever o valor de dados desconhecidos usando valores de dados relacionados e conhecidos. Para isso modela-se, matematicamente, uma **equação linear** que relaciona:

- uma variável desconhecida, ou dependente, muitas vezes chamada de **variável de resultado**.
- e uma ou mais variáveis independentes, frequentemente chamados de **preditores**, covariáveis, recursos, ou **features**.

Regressão

Housing price prediction.



Regressão

A regressão é um modelo baseado em aprendizado.

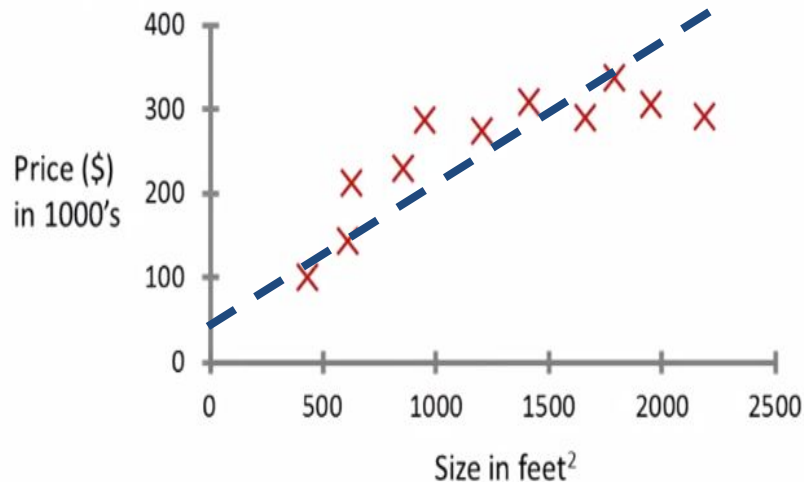
- Analizamos os dados.
- Supomos que eles seguem algum padrão.
- Utilizamos o padrão para prever dados futuros.

Regressão

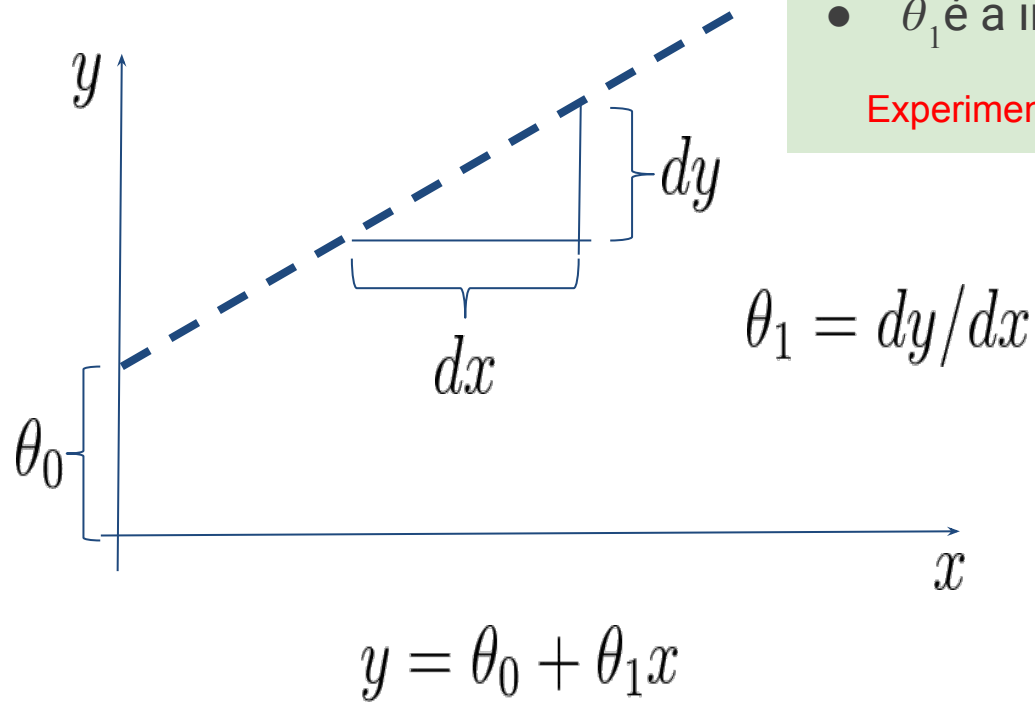
Equação da reta:

$$y = \theta_0 + \theta_1 x$$

Housing price prediction.



Regressão

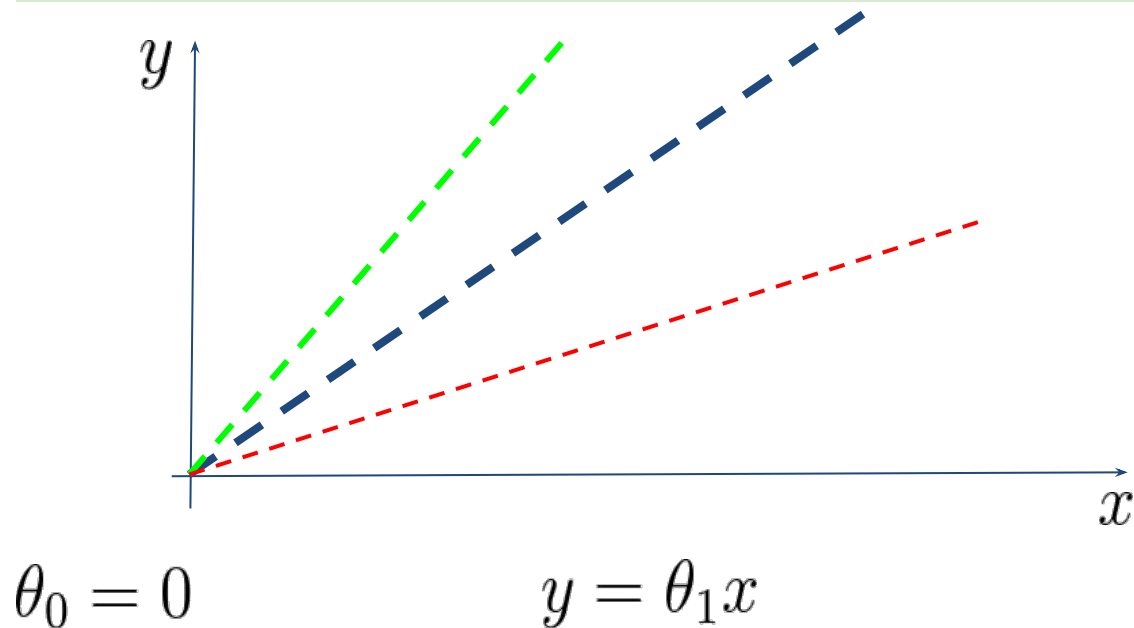


- θ_0 é o deslocamento no eixo y
- θ_1 é a inclinação da reta

Experimente algumas visualizações [aqui](#).

Regressão

- Se θ_0 é zero, a reta passa na origem



Regressão

```
import matplotlib.pyplot as plt
import numpy as np
```

```
#coeficiente angular da reta  $y = \theta_1 x + \theta_0$  (ou  $y = mx + n$ , como muitos conhecem)
 $\theta_1 = 0.8$ 
```

```
x = np.linspace(0, 1000, 1000)
#print(x)
```

```
y =  $\theta_1$  * x + 0
```

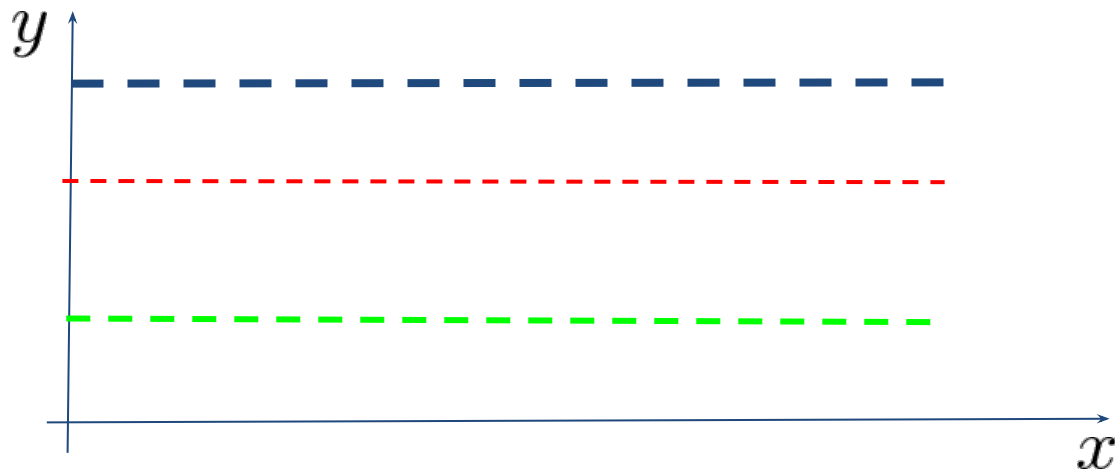
```
plt.plot(x, y, '-r')
plt.xlim(0, 1000)
plt.ylim(0, 2000)
plt.xticks(np.arange(0,1100, step=100))
```

```
plt.xlabel('Área em  $m^2$ ')
plt.ylabel('Preço em 1000\'s R$')
plt.title('Preço estimado do imóvel')
plt.grid()
plt.show()
```



Regressão

- Se θ_1 é zero, a reta será paralela ao eixo x



$$\theta_1 = 0$$

$$y = \theta_0$$

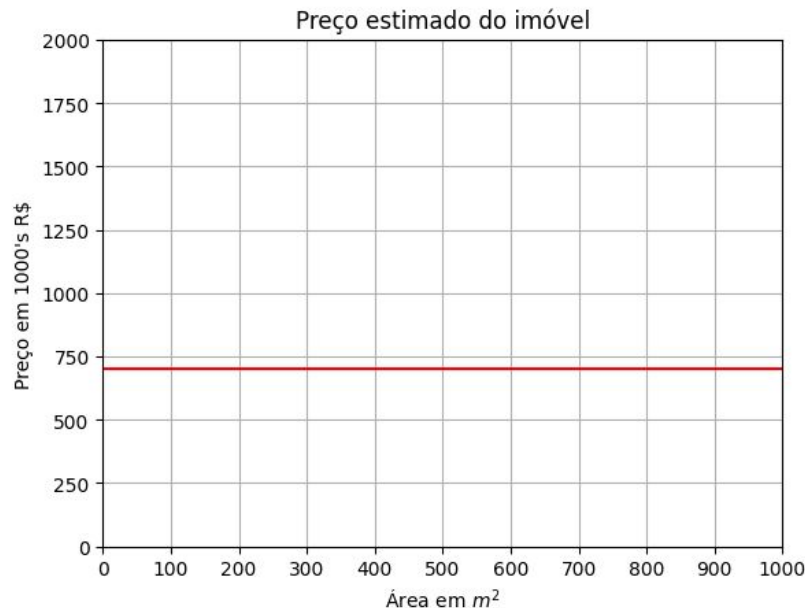
Regressão

```
import matplotlib.pyplot as plt
import numpy as np
```

```
#parâmetro para uma reta sem inclinação  $y = \theta_1 x + \theta_0$  (ou,  $y = mx + n$ )
 $\theta_0 = 700$ 
```

```
x = np.linspace(0, 1000, 1000)
y = np.ones(1000) *  $\theta_0$ 

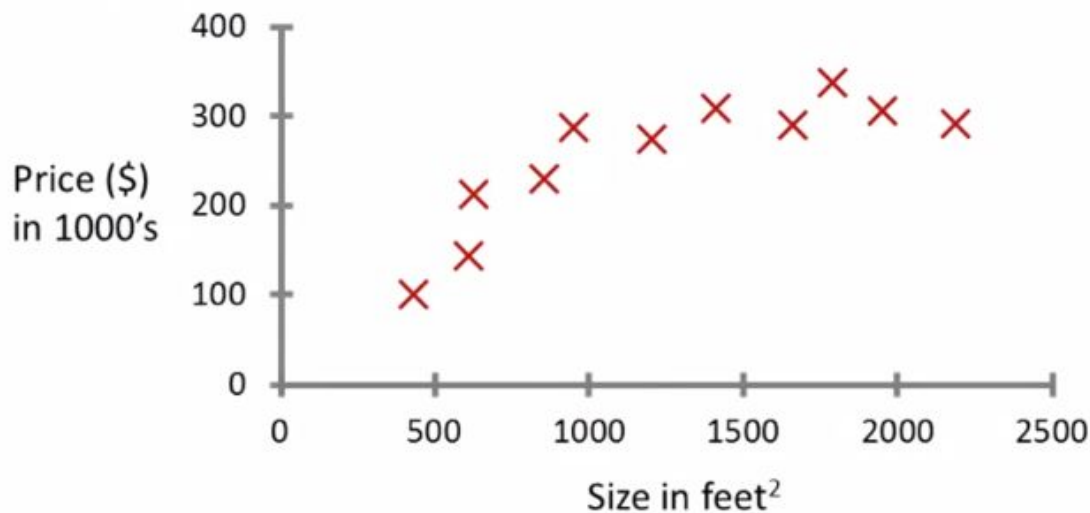
plt.plot(x, y, '-r')
plt.xlim(0, 1000)
plt.ylim(0, 2000)
plt.xticks(np.arange(0,1100, step=100))
plt.xlabel('Área em  $m^2$ ')
plt.ylabel('Preço em 1000\'s R$')
plt.title('Preço estimado do imóvel')
plt.grid()
plt.show()
```



Regressão

Voltando para o exemplo original... queremos estimar uma reta que melhor se ajuste aos pontos de observação.

Housing price prediction.



Regressão

Se a correlação linear é forte....

```
import matplotlib.pyplot as plt
import numpy as np
```

```
x = np.array([480, 510, 520, 850, 960, 1200, 1400, 1650, 1700, 1920, 2350])
y = np.array([ 92, 96.5, 98, 147.5, 164, 200, 230, 267.5, 275, 308, 372.5])
```

```
plt.plot(x, y, 'o', color='black'); #plota os pontos no gráfico
```

```
theta1 = 0.15 # inclinacao da reta
theta0 = 20   # deslocamento no eixo y
```

```
x_entrada = np.linspace(0, 2500, 2500)
y_predito = theta1 * x_entrada + theta0
```

```
plt.plot(x_entrada, y_predito, '-r')
plt.xlim(0,2500)
plt.ylim(0,400)
```

```
plt.xlabel('Área em $m^2$')
plt.ylabel('Preço em 1000\'s R$')
plt.title('Preço estimado do imóvel')
plt.grid()
plt.show()
```



Regressão

```
import matplotlib.pyplot as plt
import numpy as np
```

```
x = np.array([480, 1920])
```

```
y = np.array([ 92, 308])
```

```
plt.plot(x, y, 'o', color='black'); #plota os pontos no gráfico
```

```
# y = theta0 + theta1*x
```

```
theta1 = (y[1] - y[0])/(x[1]-x[0]) # inclinacao da reta
```

```
theta0 = y[1] - theta1 * x[1] # deslocamento no eixo y
```

```
x_entrada = np.linspace(0, 2500, 2500)
```

```
y_predito = theta1 * x_entrada + theta0
```

```
plt.plot(x_entrada, y_predito, '-r')
```

```
plt.xlim(0,2500)
```

```
plt.ylim(0,400)
```

```
plt.xlabel('Área em $m^2$')
```

```
plt.ylabel('Preço em 1000\'s R$')
```

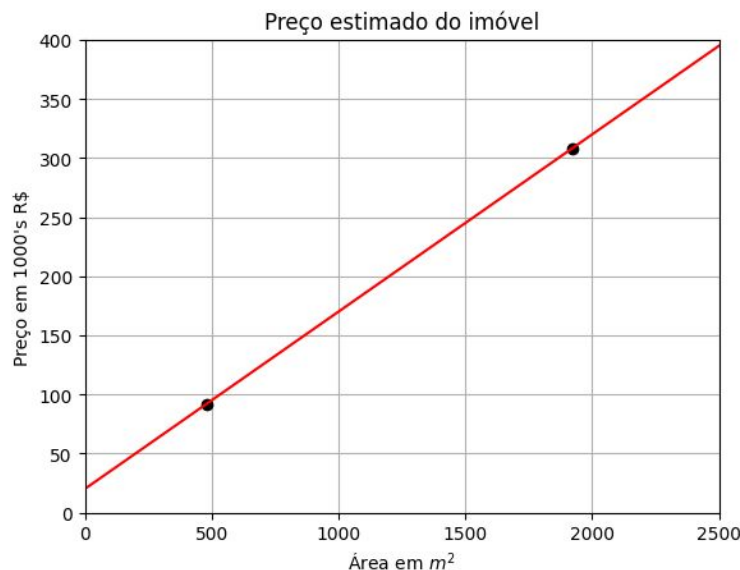
```
plt.title('Preço estimado do imóvel')
```

```
plt.grid()
```

```
plt.show()
```

Se a correlação linear é forte...

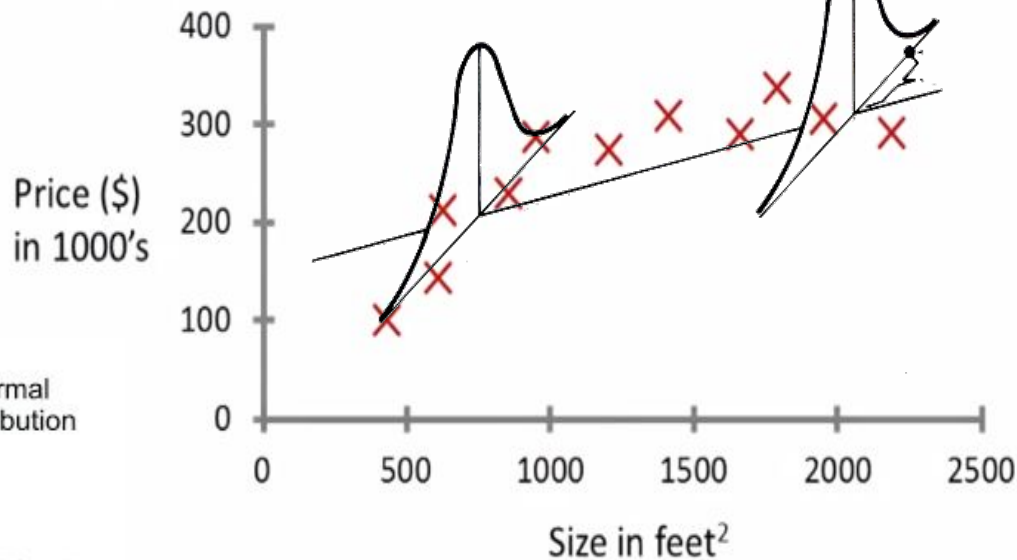
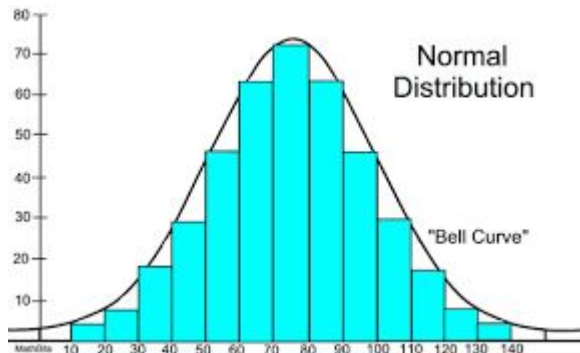
Apenas 2 pontos quaisquer são suficientes para se encontrar os valores que definem uma reta



Regressão

Mas na vida real...

Housing price prediction.



Exercício 1

Considerando os valores x e y fornecidos abaixo, tente encontrar uma reta que melhor se ajuste aos dados:

```
x = np.array([480, 510, 520, 850, 960, 1200, 1400, 1650, 1700, 1920, 2350])  
y = np.array([98, 110, 200, 210, 280, 265, 300, 287, 325, 300, 290])
```

Faça isso utilizando:

- a) Apenas θ_0
- b) Apenas θ_1
- c) Atribuindo valores para θ_0 e θ_1

Exercício

Tente ajustar, **manualmente**, uma reta que se ajuste aos dados fornecidos!

```
import matplotlib.pyplot as plt
import numpy as np
```

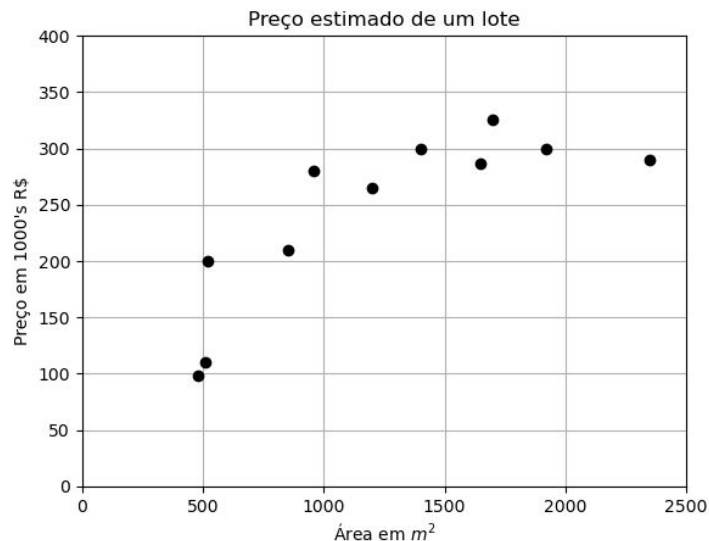
```
x = np.array([480, 510, 520, 850, 960, 1200, 1400, 1650, 1700, 1920, 2350])
y = np.array([98, 110, 200, 210, 280, 265, 300, 287, 325, 300, 290])
```

```
plt.plot(x, y, 'o', color='black');
```

```
plt.xlim(0, 2500)
```

```
plt.ylim(0, 400)
```

```
plt.xlabel('Área em $m^2$')
plt.ylabel('Preço em 1000\'s R$')
plt.title('Preço estimado de um lote')
plt.grid()
plt.show()
```



Leitura recomendada

Calculadora gráfica: [Desmos | Calculadora Gráfica](#)