Regressão logística

O que é regressão logística?

→ técnica estatística utilizada para modelar problemas de classificação

Objetivo

→ Prever a probabilidade de ocorrência de uma classe (geralmente a classe 1) → $P(y=1 \mid X)$

no caso da regressão linear, é a equação da reta

Modelo matemático

→ **função logística** (sigmoide) → A função sigmoide transforma uma equação linear em uma probabilidade entre 0 e 1

$$P(y=1 \mid X) = \frac{1}{1+e^{-(\beta_0+\beta_1 X)}}$$

Onde,

β_0 = intercepto (b)

β_1 = coeficiente angular associado a X (a)

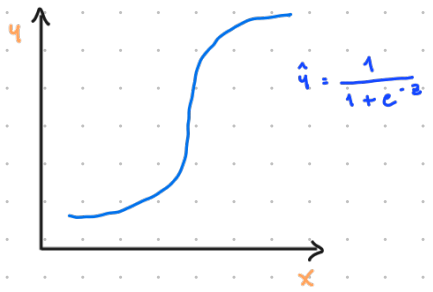
e = constante de Euler ($\sim 2{,}718$)

$P(y=1|X)$ = Probabilidade do resultado ser da classe 1, dado o valor de x

Alternativamente:

$$p = \frac{1}{1+e^{-z}}, \text{ onde } z = \beta_0 + \beta_1 X \quad (b, a)$$

→



→ **Equação do logit** (forma logarítmica)

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X$$

→ Essa transformação é chamada de logit, que converte a probabilidade em um valor que pode variar de $-\infty$ a $+\infty$, permitindo o uso de um modelo linear

Conceito	Fórmula	O que faz?
Função Logística (sigmóide)	$P = \frac{1}{1+e^{-z}}$	Converte um valor qualquer z em uma **probabilidade** entre 0 e 1
Função Logit	$\log\left(\frac{P}{1-P}\right) = z$	Converte uma **probabilidade** de volta para o valor z (score linear)

→ Para prever a probabilidade

→ forma linear do modelo

→ A regressão linear é um modelo linear no logit, mas a saída é não linear graças à sigmoide. Ela aprende ajustando os coeficientes para minimizar o erro de predição probabilística

Resumo do fluxo de aprendizado

I) Definir a equação linear (modelo base)

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_m x_m$$

II) Aplicar a função sigmoide (função logística)

$$\hat{y} = \frac{1}{1 + e^{-z}}$$

↳ isso transforma o valor de z em uma probabilidade entre 0 e 1

III) Comparar a previsão com o valor real

Para cada exemplo no treino:

i) valor real: $y \in \{0,1\}$

ii) valor previsto: $\hat{y} \in [0,1]$

IV) Calcular o erro usando a função de custo (Log loss)

$$J(\theta) = -\frac{1}{n} \sum [y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})]$$

↳ Mede a distância entre a probabilidade prevista e o valor real, penalizando com mais força as previsões muito erradas

V) Otimizar os parâmetros via gradiente descendente

$$\beta_j = \beta_j - \alpha \cdot \frac{\partial J}{\partial \beta_j}$$

Se a inclinação está para cima, a reta tem que diminuir o β, vice-versa

β_j → coefic. a ser ajustado ↳ Angular ou linear

α → taxa de aprendizado ↳ Step size

$\frac{\partial J}{\partial \beta_j}$ → Derivada da função de custo com relação ao coeficiente ↳ inclinação da reta tangente em um determinado ponto

↳ Ajusta os coeficientes em direção ao mínimo

VI) Repetir (treinar por várias iterações)

↳ Recalcular $\hat{y}$, custo e gradientes

↳ Atualizar os coeficientes β.

↳ Parar quando:

- A função de custo converge (diferença pequena entre épocas)
- Atinge o número máximo de iterações

VI) Repetir (treinar por várias iterações)

↳ Entrada nova X_{nova} → calcula z

↳ Aplica sigmoide → gera y

↳ Se:

$\hat{y} \geq 0.5$ → classe 1

$\hat{y} \leq 0.5$ → classe 0

↳ considerando o threshold padrão