

data

Aula 2

Enzo Tonon Morente,

/EnzoTM

Presença

- Linktree: Presente na bio do nosso instagram
- Presença ficará disponível até 1 hora antes da próxima aula
- É necessário 70% de presença para obter o certificado



Presença e Github





Revisão



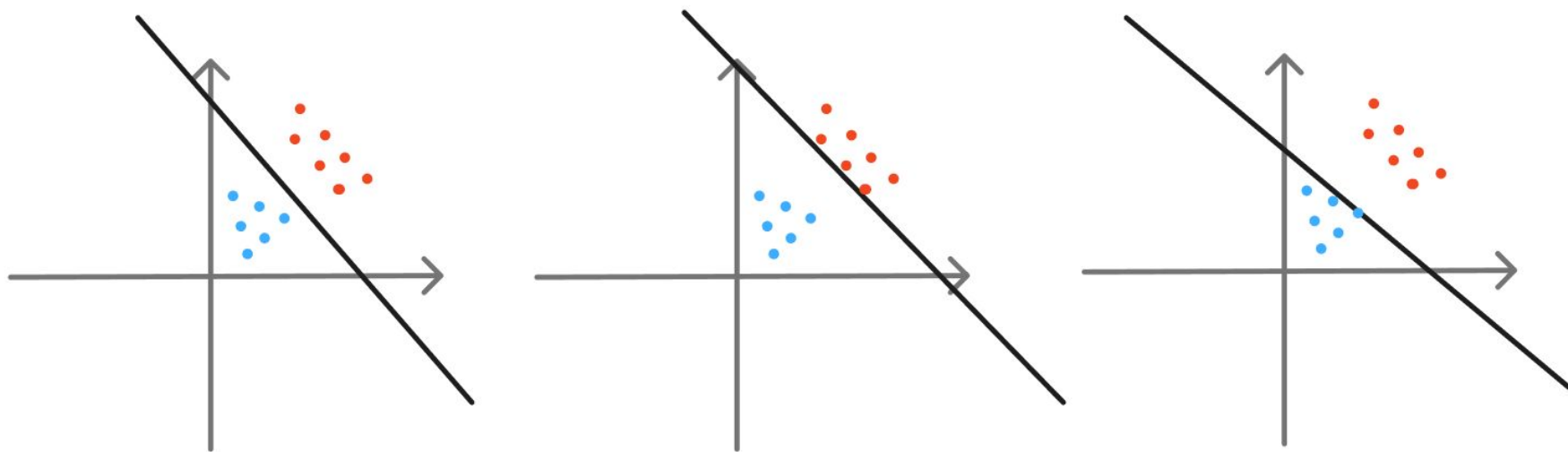


Decision Boundary

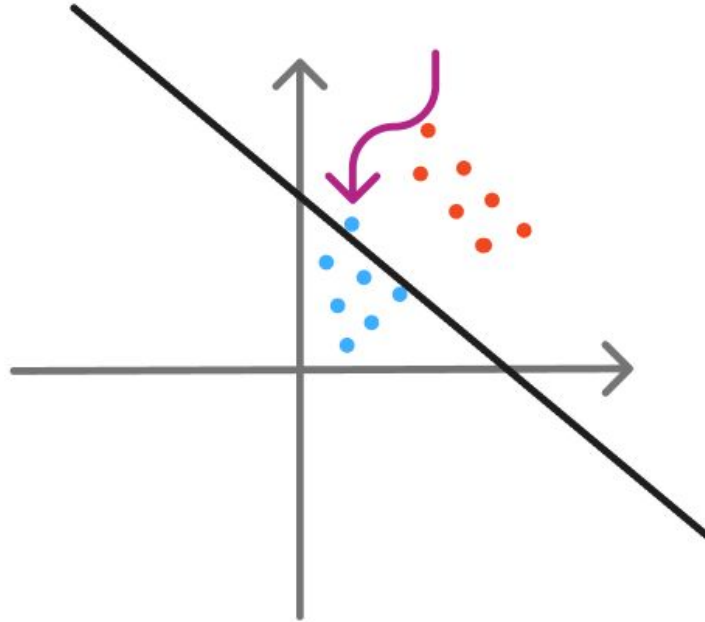


Decision Boundary

- O Perceptron nos dá a melhor decision boundary?

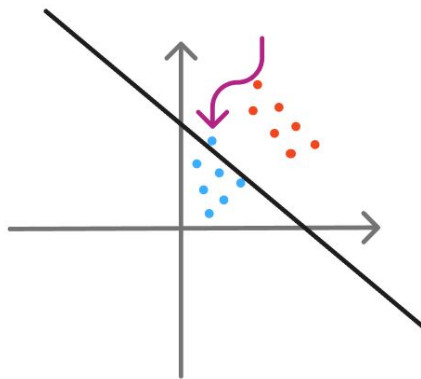


Decision Boundary



Decision Boundary

- Queremos deixar uma margem na decision boundary do modelo
- Essa margem irá servir para generalização de dados ainda não vistos

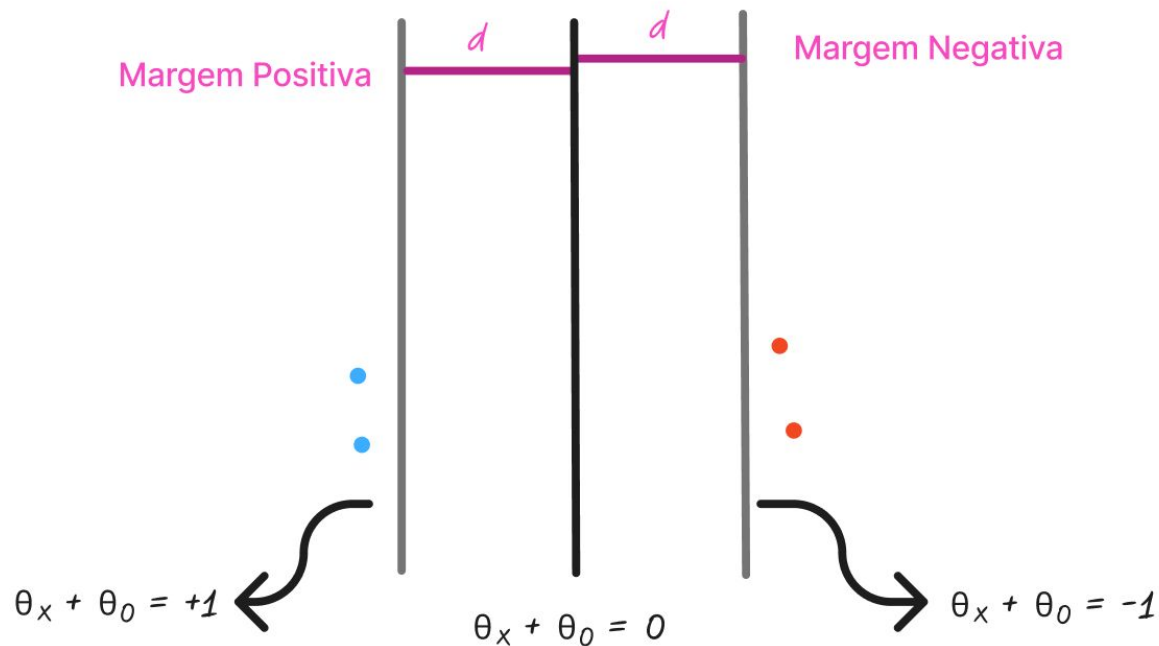




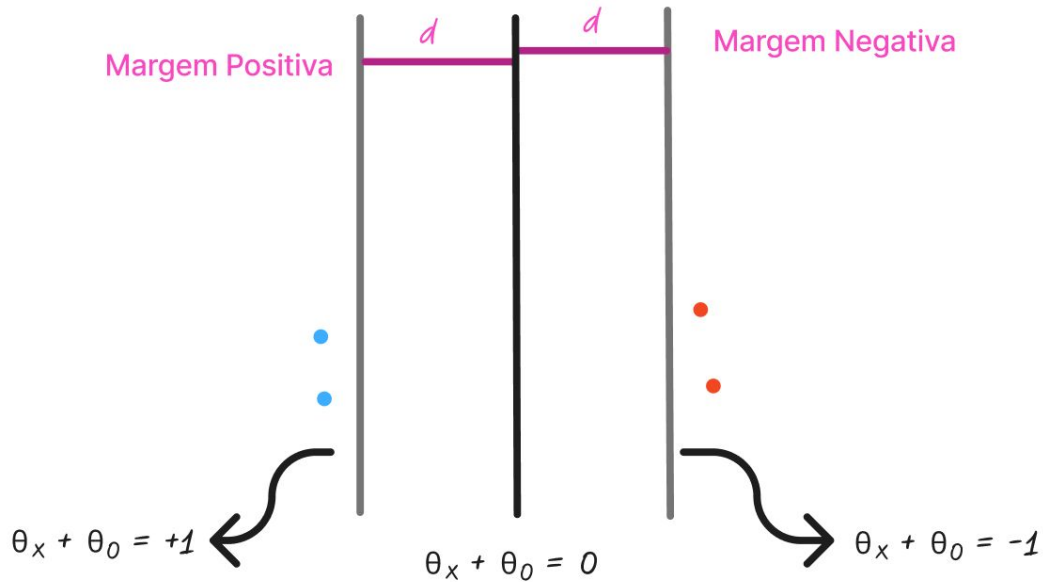
Margin Boundary



Definição Decision Boundary



Definição Decision Boundary



$$d = \frac{1}{||\vec{\theta}||}$$



Função Gamma



Função Gamma

- Se o ponto foi corretamente classificado
- Se o ponto está dentro ou fora da margem



Função Gamma

- Se o ponto foi corretamente classificado

$$y_i(\theta \cdot X_i + \theta_0) \leq 0$$

- Se o ponto está dentro ou fora da margem



Função Gamma

- Se o ponto foi corretamente classificado

$$y_i(\theta \cdot X_i + \theta_0) \leq 0$$

- Se o ponto está dentro ou fora da margem

$$d(\vec{x}_i, \theta \cdot \vec{x} + \theta_0 = 0) > \frac{1}{\|\vec{\theta}\|}$$



Relembrando

Ponto: (x_1, x_2)



Relembrando

Ponto: (x_1, x_2)

Reta: $(A_1x_1 + A_2x_2 + B = 0)$



Relembrando

Ponto: (x_1, x_2)

Reta: $(A_1x_1 + A_2x_2 + B = 0)$

Distância do ponto a reta: $\frac{|\vec{A} \cdot \vec{x} + B|}{\|\vec{A}\|}$



Relembrando

Ponto: (x_1, x_2)

Reta: $(A_1x_1 + A_2x_2 + B = 0)$

Distância do ponto a reta: $\frac{|\vec{A} \cdot \vec{x} + B|}{\|\vec{A}\|}$

Ponto: (x_1, x_2)



Relembrando

Ponto: (x_1, x_2)

Reta: $(A_1x_1 + A_2x_2 + B = 0)$

Distância do ponto a reta: $\frac{|\vec{A} \cdot \vec{x} + B|}{\|\vec{A}\|}$

Ponto: (x_1, x_2)

Reta: $\vec{\theta} \cdot \vec{x} + \theta_0 = 0$



Relembrando

Ponto: (x_1, x_2)

Reta: $(A_1x_1 + A_2x_2 + B = 0)$

Distância do ponto a reta: $\frac{|\vec{A} \cdot \vec{x} + B|}{\|\vec{A}\|}$

Ponto: (x_1, x_2)

Reta: $\vec{\theta} \cdot \vec{x} + \theta_0 = 0$

Distância do ponto a reta: $\frac{|\theta \cdot x + \theta_0|}{\|\vec{\theta}\|}$



Construindo função Gamma

Distância: $\frac{|\theta \cdot x + \theta_0|}{\|\vec{\theta}\|}$

Numerador sempre positivo



Construindo função Gamma

Distância: $\frac{|\theta.x + \theta_0|}{\|\vec{\theta}\|}$

Loss: $y_i(\theta.x_i + \theta_0)$

Numerador sempre positivo

É capaz de dizer se houve um erro ou acerto



Construindo função Gamma

Distância: $\frac{|\theta.x + \theta_0|}{\|\vec{\theta}\|}$

Loss: $y_i(\theta.x_i + \theta_0)$

$$\gamma(\theta, \theta_0) = \frac{y_i(\theta.x_i + \theta_0)}{\|\vec{\theta}\|}$$



Função Gamma

$$\gamma(\theta, \theta_0) = \frac{y_i(\theta \cdot x_i + \theta_0)}{\|\vec{\theta}\|}$$

$\gamma(\theta, \theta_0) > 0$, exemplo corretamente classificado

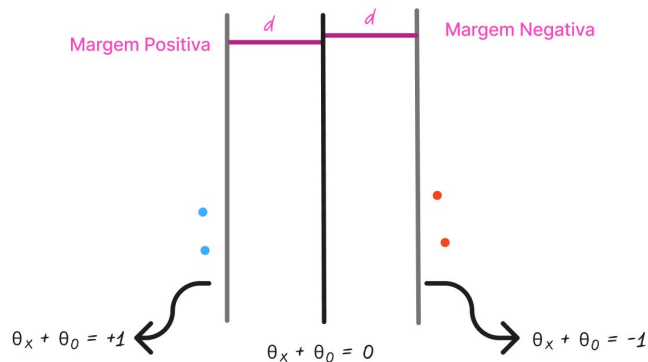
$\gamma(\theta, \theta_0) < 0$, exemplo classificado errado

$|\gamma(\theta, \theta_0)| > \frac{1}{\|\theta\|}$, exemplo fora da margem

$0 < |\gamma(\theta, \theta_0)| < \frac{1}{\|\theta\|}$, exemplo dentro da margem



Resumindo



Queremos deixar uma
margem para o modelo
generalizar

$$\gamma(\theta, \theta_0) = \frac{y_i(\theta \cdot x_i + \theta_0)}{\|\vec{\theta}\|}$$

Aprendemos a quantificar o
acerto/erro e a distância do
ponto em relação a margem





Loss



Nova Loss

Loss da aula passada quantificava
erro ou acerto

$$Loss = y_i(\theta x + \theta_0) \leq 0$$

- A nova Loss deve ser capaz de quantificar:
 - erro ou acerto
 - fora ou dentro da margem
 - se erro, a magnitude do erro (distância)

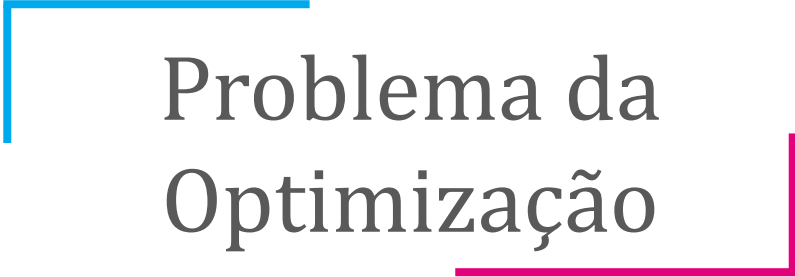


Hinge Loss

$$Loss_h(z) = \begin{cases} 0 & \text{if } z \geq 1, \\ 1 - z & \text{if } z < 1. \end{cases}$$

$$z = y^{(i)} \cdot (\theta \cdot x^{(i)} + \theta_0)$$





Problema da Optimização



Optimização

- A nossa Loss quantifica a quantidade de erros e o quão errado está cada um
- Queremos fazer com que a Loss seja a mínima possível

$$\min(Loss)$$



Optimização

- Queremos fazer com que a Loss seja a mínima possível

$$\min\left(\begin{cases} 0 & \text{if } z \geq 1, \\ 1 - z & \text{if } z < 1. \end{cases}, \text{ onde } z = y^{(i)} \cdot (\theta \cdot x^{(i)} + \theta_0)\right)$$



Optimização

- Queremos fazer com que a Loss seja a mínima possível

$$\min\left(\begin{cases} 0 & \text{if } z \geq 1, \\ 1 - z & \text{if } z < 1. \end{cases}, \text{ onde } z = y^{(i)} \cdot (\theta \cdot x^{(i)} + \theta_0)\right)$$

- Isso não é o suficiente



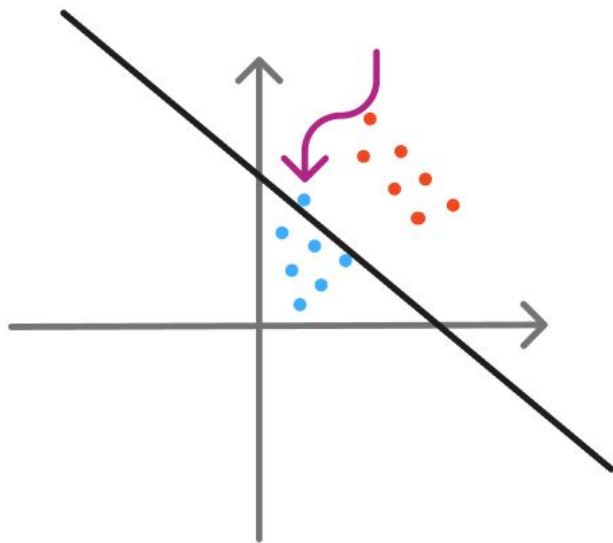
O que estamos pedindo pro modelo

- Estamos pedindo para que o modelo diminua o erro durante o treinamento
- Quantificamos para o modelo que o erro agora deve levar em consideração a magnitude
- Não estamos levando em consideração que queremos ter uma margem



Decision Boundary

- Melhoramos a forma de quantificar o erro, não a generalização





Regularização



Regularização

- Técnica que nos permita expandir a capacidade de generalização de nosso modelo
- Fazer com que o modelo não fique hiper fixado nos dados de treinamento
- Como fazer isso?



Regularização

- A regularização está totalmente ligada a margem
- Para fazer o modelo generalizar queremos que o mesmo tenha uma certa margem
- Temos que favorecer margens grandes



Regularização

- Quanto maior o valor de theta, menor a distância à margem
- Quanto menor o valor de theta, maior a distância à margem
- Queremos beneficiar margens ‘grandes’
 - Capacidade de generalizar

$$d = \frac{1}{||\vec{\theta}||}$$



Regularização

- Queremos beneficiar margens ‘grandes’
 - Capacidade de generalizar

$$\max\left(\frac{1}{\|\theta\|}\right)$$

$$d = \frac{1}{\|\vec{\theta}\|}$$



Novo Objetivo

$$\min(Loss) \text{ and } \max\left(\frac{1}{\|\theta\|}\right)$$



Novo Objetivo

$$\max\left(\frac{1}{\|\theta\|}\right) = \min(\|\theta\|)$$



Novo Objetivo

$$\min(\|\theta\|) = \min(\frac{1}{2}\|\theta\|^2)$$



Novo Objetivo

$$\min(Loss) \text{ and } \min(\frac{1}{2}\|\theta\|^2)$$



Função Objetivo

Queremos minimizar a seguinte função objetivo

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^n \text{Loss}_h(y^{(i)}(\theta.x + \theta_0)) + \frac{\lambda}{2} \|\theta\|^2$$

A genialidade de minimizar essa função é que minimizar J significa minimizar a loss e θ , e, minimizar θ significa aumentar a margem



Função Objetivo

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^n \text{Loss}_h(y^{(i)}(\theta.x + \theta_0)) + \frac{\lambda}{2} \|\theta\|^2$$

- Leva em consideração a Loss
- Leva em consideração ter margens grandes
- Quem é lambda?



Lambda

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^n Loss_h(y^{(i)}(\theta.x + \theta_0)) + \frac{\lambda}{2} \|\theta\|^2$$

- Lambda é o termo de regularização
- Define o grau de importância que theta tem em J



Lambda

$$J(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^n Loss_h(y^{(i)}(\theta.x + \theta_0)) + \frac{\lambda}{2} \|\theta\|^2$$

- Lambda alto: importância dada à se ter uma margem alta
- Lambda baixo: importância dada à se ter uma Loss baixa



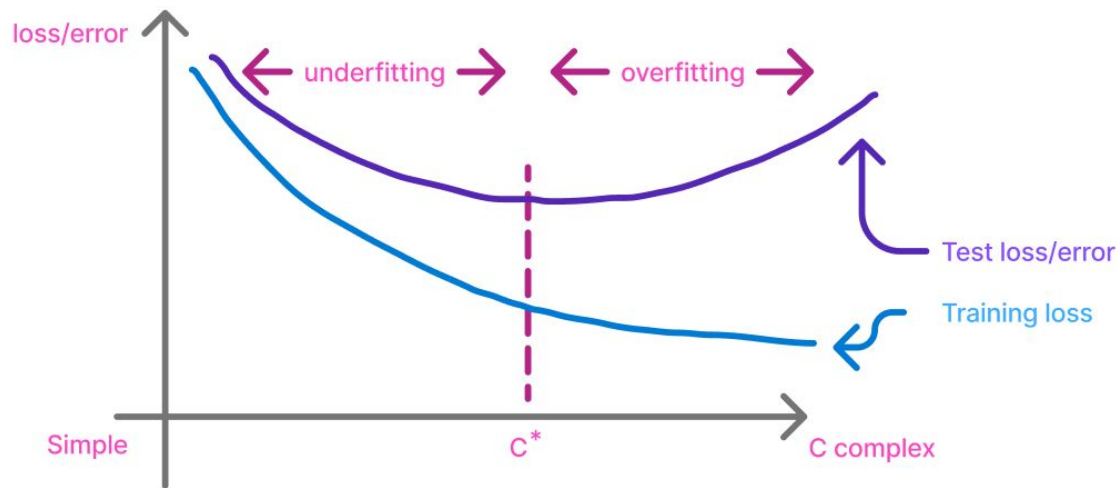
Lambda

A complexidade do nosso modelo é definida por: $C = \frac{1}{\lambda}$

- Lambda alto -> alta margem: baixa complexidade
- Lambda baixo -> Loss baixa: alta complexidade
 - Modelo muito complexo é aquele que está super ajustado aos dados de treinamento
 - Chamamos isso de Overfitting



Lambda

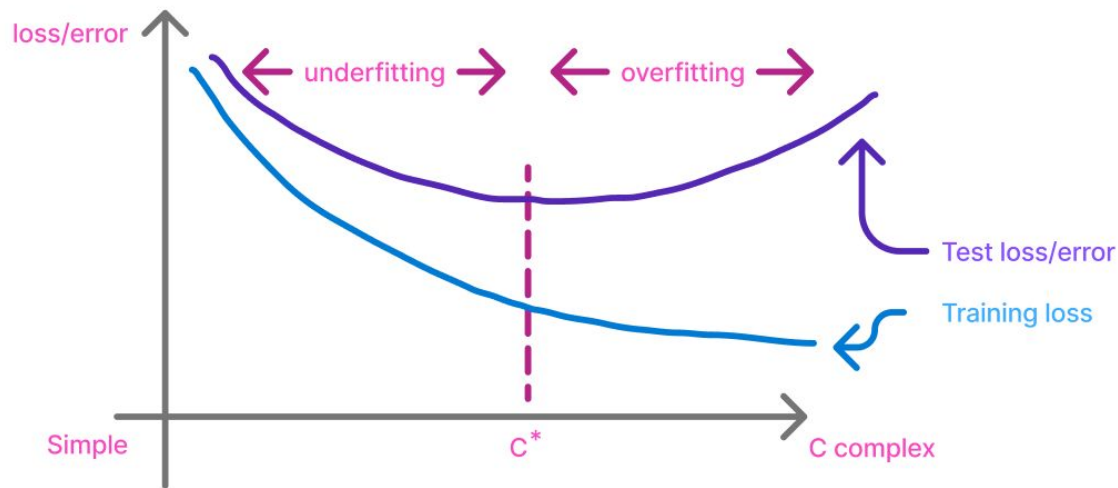


Overfitting: decora os dados de treinamento, invés de aprender padrões

Underfitting: muito simples e não consegue capturar a complexidade dos dados



Lambda



Melhor valor de Lambda: validação





Gradiente Discendente



Gradiente Descendente

- Gradiente Descendente é a forma que iremos utilizar para minimizar a nossa função objetivo
- Para falarmos sobre Gradiente precisamos primeiro introduzir a noção de derivada

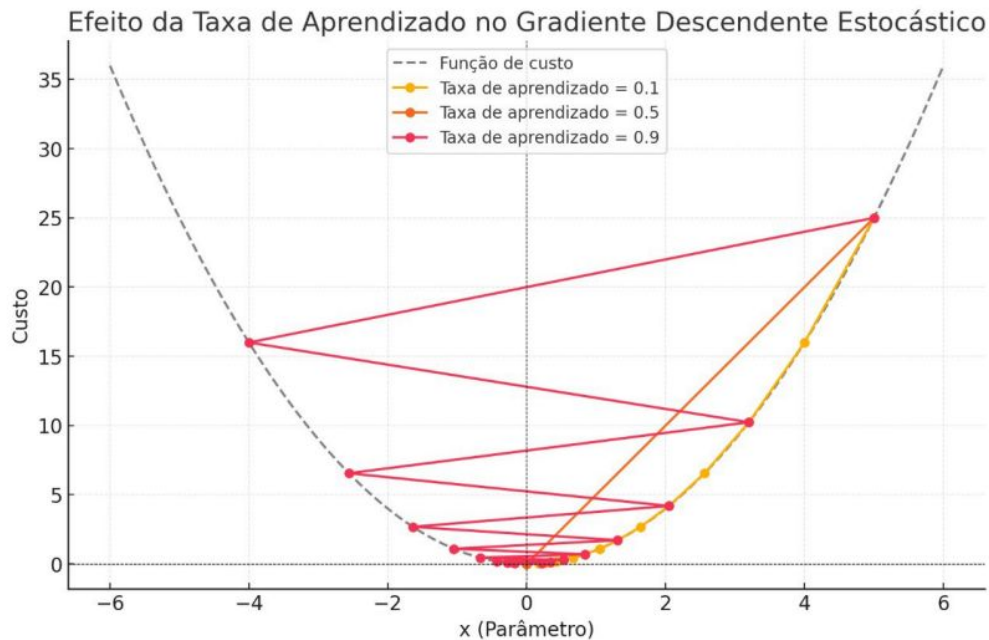


Derivada

- Taxa de variação de uma função em um certo ponto
- Andar na direção da derivada:
 - aumentar o valor da função
- Andar contrário da derivada:
 - diminuir o valor da função



Learning Rate



Gradiente Descendente

- Vimos que para minimizar uma função devemos ir ao sentido oposto da função com uma certa learning rate

$$\theta = \theta - \eta \nabla_{\theta} [J]$$

$$\theta_0 = \theta_0 - \eta \nabla_{\theta_0} [J]$$



Stochastic Gradiente Descendente

1. $\theta = \text{valor aleatório}; \theta_0 = \text{valor aleatório}$
2. for t in range(T)
3. sortear i em: $\{x_0, x_1, \dots, x_n\}$
4. $\theta = \theta - \eta \nabla_{\theta}[J]$
5. $\theta_0 = \theta_0 - \eta \nabla_{\theta_0}[J]$



Calculando Gradiente

$$\nabla_{\theta} J = \begin{cases} 0, & \text{if loss } 0, \\ -y_i x_i, & \text{if loss } < 0 \end{cases} + \lambda \theta$$

$$\nabla_{\theta_0} J = \begin{cases} 0, & \text{if loss } 0, \\ -y_i, & \text{if loss } < 0 \end{cases}$$



Stochastic Gradiente Descendente

1. $\theta = \text{valor aleatório}; \theta_0 = \text{valor aleatório}$
2. for t in range(T)
3. sortear i em: $\{x_0, x_1, \dots, x_n\}$
4. if $\text{Loss} < 0$:
 4. $\theta = \theta + \eta \cdot (x_i y_i) - \eta(\lambda \theta)$
 5. $\theta_0 = \theta_0 + \eta \cdot y_i$
6. else :
 7. $\theta = \theta - \eta(\lambda \theta)$





Bag of Words



Bag of Words

	the	red	dog	cat	eats	food
1. the red dog →	1	1	1	0	0	0
2. cat eats dog →	0	0	1	1	1	0
3. dog eats food →	0	0	1	0	1	1
4. red cat eats →	0	1	0	1	1	0



data@icmc.usp.br



@data.icmc



/c/DataICMC



/icmc-data



data.icmc.usp.br



obrigado por sua
presença!

