Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina

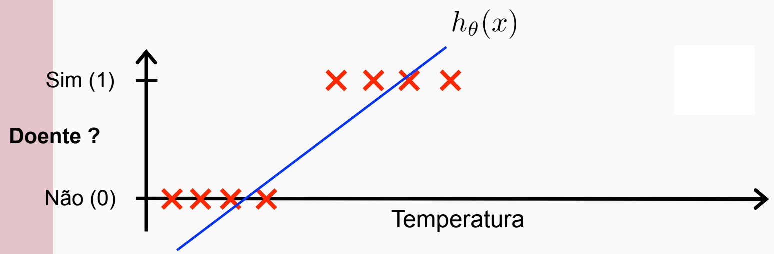
Regressão Logística

Prof. Tiago A. Almeida

Classificação

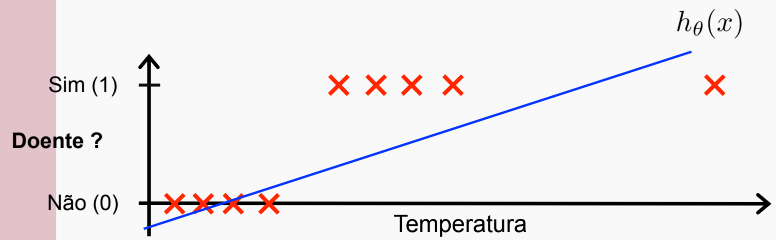
- Rótulo discreto (classe)
  - Normalmente: $y \in \{0, 1\}$
    - **0**: Classe negativa (email legítimo, saudável)
    - **1**: Classe positiva (email spam, doente)

Classificação



- Classificador:
  - Se $h_\theta(x) \geq 0.5$, classe (y) = 1
  - Se $h_\theta(x) < 0.5$, classe (y) = 0

Classificação



- Classificador:
  - Se $h_\theta(x) \geq 0.5$, classe (y) = 1
  - Se $h_\theta(x) < 0.5$, classe (y) = 0

Regressão Logística

- Na classificação $y \in \{0, 1\}$
- Na regressão $h_\theta(x)$ pode ser > 1 ou < 0
- Na regressão logística: $0 \leq h_\theta(x) \leq 1$
  - Como ?

Regressão Logística

- Na classificação $y \in \{0, 1\}$
- Na regressão $h_\theta(x)$ pode ser > 1 ou < 0
- Na regressão logística: $0 \leq h_\theta(x) \leq 1$
  - Como ?
  - **Alterando a representação da hipótese**

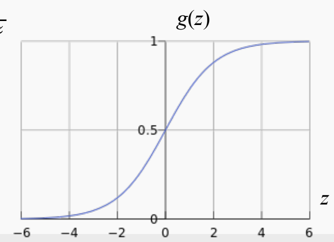
Regressão Logística - Hipótese

- Na regressão linear: $h_\theta(x) = \theta^T x$
- Entretanto, deseja-se que $0 \leq h_\theta(x) \leq 1$
- Para isso, é necessário usar **função logística *g*(*x*)** (ou sigmoidal)

$$h_\theta(x) = g(\theta^T x)$$

Regressão Logística - Hipótese

- **Função logística**
  - Hipótese: $h_\theta(x) = g(\theta^T x)$
  - Mapeamento: $g(z) = \frac{1}{1+e^{-z}}$



Regressão Logística - Hipótese

- **Função logística**
  - Hipótese: $h_\theta(x) = g(\theta^T x)$
  - Mapeamento: $g(z) = \frac{1}{1+e^{-z}}$
  - Portanto: $h_\theta(x) = \dfrac{1}{1+e^{-\theta^T x}}$

Regressão Logística - Hipótese

- **Hipótese:** $h_\theta(x) = \dfrac{1}{1+e^{-\theta^T x}}$
- $h_\theta(x)$ = probabilidade da amostra pertencer à classe **y = 1**, dado que ela possui os atributos **x**

$$P(y=0|x;\theta) + P(y=1|x;\theta) = 1$$
$$P(y=0|x;\theta) = 1 - P(y=1|x;\theta)$$

Regressão Logística - Hipótese

- **Hipótese:** $h_\theta(x) = \dfrac{1}{1+e^{-\theta^T x}}$
- $h_\theta(x)$ = probabilidade da amostra pertencer à classe **y = 1**, dado que ela possui os atributos **x**
  - Se $h_\theta(x)$ = 0.9 para classificação de paciente, então conclui-se que existe 90% de chance do paciente estar doente

Limite de decisão

- Regressão logística

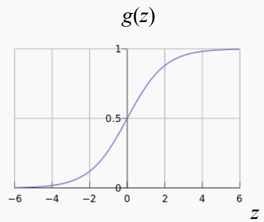
$h_\theta(x) = g(\theta^T x)$ $\qquad g(z) = \frac{1}{1+e^{-z}}$

- Suponha que $y = 1$, se $h_\theta(x) \geq 0.5$

$$\theta^T x \geq 0$$

- Suponha que $y = 0$, se $h_\theta(x) < 0.5$

$$\theta^T x < 0$$



Função Custo

- Treinamento: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$

$$x \in \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{bmatrix} \qquad x_0 = 1, y \in \{0, 1\}$$

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

- Como escolher os parâmetros θ?

Função Custo

- Regressão linear: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

Função Custo

- Regressão linear: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$
- Regressão logística: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$

Função Custo

- Regressão linear: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$
- Regressão logística: $J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$

$$\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)}) = \frac{1}{2} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

Função custo não é convexa!

Função Custo

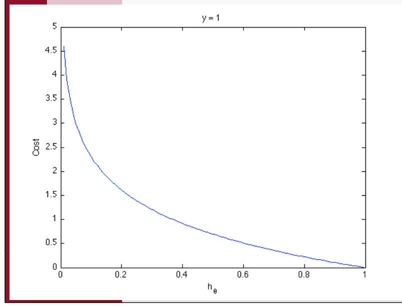
- Regressão logística: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{se } y = 1 \\ -\log(1 - h_\theta(x)) & \text{se } y = 0 \end{cases}$$

Função Custo

- Regressão logística: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{se } y = 1 \\ -\log(1 - h_\theta(x)) & \text{se } y = 0 \end{cases}$$



$Cost = 0 \quad se \quad y = 1, \quad h_\theta(x) = 1$

$h_\theta(x) \to 0 \quad Cost \to \inf$

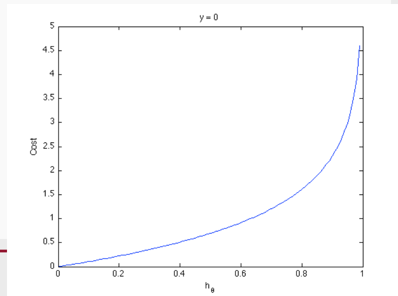
Função Custo

- Regressão logística: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{se } y = 1 \\ -\log(1 - h_\theta(x)) & \text{se } y = 0 \end{cases}$$

$Cost = 0 \quad se \quad y = 0, \quad h_\theta(x) = 0$

$h_\theta(x) \to 1 \quad Cost \to \inf$



Função Custo simplificada

- Regressão logística: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{se } y = 1 \\ -\log(1 - h_\theta(x)) & \text{se } y = 0 \end{cases}$$

- Como *y = 1* ou *y = 0*, é possível simplificar a função *Cost* para:

$$Cost(h_\theta(x), y) = -y \;\log(h_\theta(x)) - (1 - y) \;\log(1 - h_\theta(x))$$

Gradiente descendente

- **Regressão logística**

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$
$$= -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log (1-h_\theta(x^{(i)}))]$$

- **Objetivo: ajustar θ**

$$\min_\theta J(\theta)$$

- **Classificador (dado novo x):** $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

Gradiente descendente

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log (1-h_\theta(x^{(i)}))]$$

- Deseja-se $\min_\theta J(\theta)$
- **Método do Gradiente:**

Repita $\{$

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$\}$ (atualização simultânea para todo θ_j)

Gradiente descendente

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log (1-h_\theta(x^{(i)}))]$$

- Deseja-se $\min_\theta J(\theta)$
- **Método do Gradiente:**

Parece idêntico à Regressão Linear

Repita $\{$

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

$\}$ (atualização simultânea para todo θ_j)

Gradiente descendente

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log (1-h_\theta(x^{(i)}))]$$

- Deseja-se $\min_\theta J(\theta)$
- **Método do Gradiente:**

$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

Repita $\{$

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

$\}$ (atualização simultânea para todo θ_j)

Funções Avançadas de Otimização

- No Matlab/Octave existem diversas funções de otimização que desenvolvem o papel do GD de forma muito mais eficiente
- **Exemplos**
  - Gradiente descendente
  - Gradiente conjugado
  - BFGS
  - L-BFGS

Funções Avançadas de Otimização

- **Vantagens**
  - Normalmente são mais rápidas
    - Necessárias em programas de grande porte
  - Não há necessidade de ajustar manualmente o valor do passo α
- **Desvantagem**
  - Maior complexidade de interpretação e implementação

Funções Avançadas de Otimização

- **Exemplo**

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

$$J(\theta) = (\theta_1 - 5)^2 + (\theta_2 - 5)^2$$

$$\frac{\partial}{\partial \theta_1} J(\theta) = 2(\theta_1 - 5)$$

$$\frac{\partial}{\partial \theta_2} J(\theta) = 2(\theta_2 - 5)$$

Funções Avançadas de Otimização

- **Exemplo**

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

$$J(\theta) = (\theta_1 - 5)^2 + (\theta_2 - 5)^2$$

$$\frac{\partial}{\partial \theta_1} J(\theta) = 2(\theta_1 - 5)$$

$$\frac{\partial}{\partial \theta_2} J(\theta) = 2(\theta_2 - 5)$$

```
function [J, gradiente]
            = funcaoCusto(theta)
  J = (theta(1)-5)^2 + ...
       (theta(2)-5)^2;
  gradiente= zeros(2,1);
  gradiente(1)= 2*(theta(1)-5);
  gradiente(2)= 2*(theta(2)-5);
```

Funções Avançadas de Otimização

- **Exemplo**

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$$

$$J(\theta) = (\theta_1 - 5)^2 + (\theta_2 - 5)^2$$

$$\frac{\partial}{\partial \theta_1} J(\theta) = 2(\theta_1 - 5)$$

$$\frac{\partial}{\partial \theta_2} J(\theta) = 2(\theta_2 - 5)$$

```
function [J, gradiente]
            = funcaoCusto(theta)
  J = (theta(1)-5)^2 + ...
      (theta(2)-5)^2;
  gradiente= zeros(2,1);
  gradiente(1)= 2*(theta(1)-5);
  gradiente(2)= 2*(theta(2)-5);
```

```
opcoes = optimset('GradObj', 'on', 'MaxIter', '100');
ThetaInicial = zeros(2,1);
[ThetaOtimo, Custo_J, exitFlag] ...
     = fminunc(@funcaoCusto, ThetaInicial, opcoes);
```


Funções Avançadas de Otimização

- **Implementação**

theta = $\begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$

```
function [J, gradiente] = funcaoCusto(theta)
  J = [código para calcular J(θ) ];
  gradiente(1)= [código para calcular ∂/∂θ0 J(θ) ];
  gradiente(2)= [código para calcular ∂/∂θ1 J(θ)];
     ⋮
  gradiente(n+1)= [código para calcular ∂/∂θn J(θ)];
```

Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina

Regularização

Prof. Tiago A. Almeida

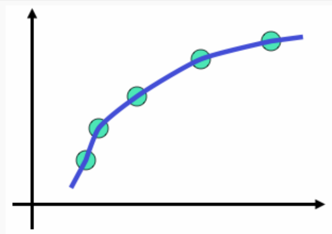
Overfitting

- Em problemas com muitos atributos pode ocorrer:
  - especialização nos dados de treinamento
  - desempenho ruim em novos dados de entrada
    - problema na “generalização”

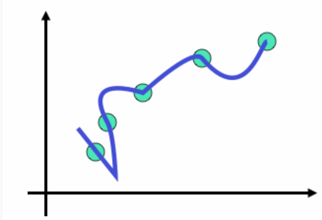
Overfitting

- **Possíveis soluções**
  - **Reduzir a quantidade de atributos**
    - Seleção manual dos atributos que deverão ser mantidos
    - Usar algoritmo para seleção automática de atributos
  - **Regularização**
    - Manter todos os atributos, mas reduzir o valor de cada parâmetro θ_j
    - Funciona bem quando há uma grande quantidade de atributos, cada um contribuindo um pouco para a predição final y

Regularização: regressão linear

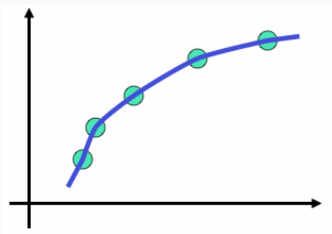


$\theta_0 + \theta_1 x + \theta_2 x^2$

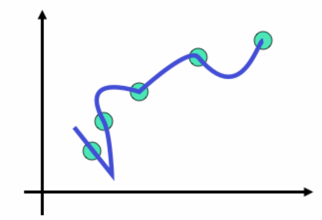


$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Regularização: regressão linear



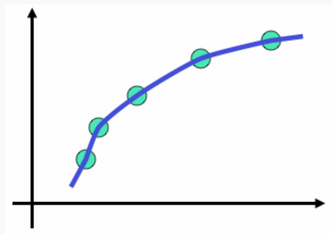
$\theta_0 + \theta_1 x + \theta_2 x^2$



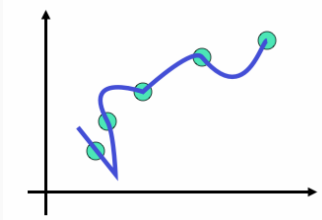
$\theta_0 + \theta_1 x + \theta_2 x^2 + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

Deseja-se reduzir a influência de θ_3 e θ_4

Regularização: regressão linear



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

Opção: penalizar θ_3 e θ_4

$$\min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000.\theta_3^2 + 1000.\theta_4^2$$

Regularização: regressão linear

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \underbrace{\lambda\sum_{j=1}^{n}\theta_j^2}\right]$$

Parâmetro de regularização

Regularização: regressão linear

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

- Se λ for muito grande (10^{10}), então

$$\theta_j \approx 0$$

- Portanto, haverá *underfitting*

Regularização: regressão logística

$$J(\theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)}\log h_\theta(x^{(i)}) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

- **Gradiente descendente**

Repita $\{$

$$\theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j\right]$$

$\}$

$(j = \cancel{0}, 1, 2, 3, \ldots, n)$

Regularização: regressão logística

- **Função avançada de otimização**

```
function [J, gradiente] = funcaoCusto(theta)
```

`J = [`código para calcular $J(\theta)$ `];`

$$J(\theta) = \left[-\frac{1}{m}\sum_{i=1}^{m} y^{(i)}\log\left(h_\theta(x^{(i)}) + (1-y^{(i)})\log 1 - h_\theta(x^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

`gradiente(1)= [`código para calcular $\frac{\partial}{\partial\theta_0}J(\theta)$ `];`

$$\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}$$

`gradiente(2)= [`código para calcular $\frac{\partial}{\partial\theta_1}J(\theta)$`];`

$$\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_1^{(i)} + \frac{\lambda}{m}\theta_1$$

`gradiente(3)= [`código para calcular $\frac{\partial}{\partial\theta_2}J(\theta)$`];`

$$\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_2^{(i)} + \frac{\lambda}{m}\theta_2$$

$\vdots$

`gradiente(n+1)= [`código para calcular $\frac{\partial}{\partial\theta_n}J(\theta)$`];`