Aula 02 - Modelos Lineares

Clodoaldo A. M. Lima

19 de agosto de 2021

Programa de Pós-Graduação em Sistemas de Informação
Mestrado acadêmico - EACH - USP
http://ppgsi.each.usp.br

Conteúdo

- Modelos lineares
  - Classificação
  - Regressão
  - Regressão logistica
- Seleção de modelos
- No Free Lunch Theorem

Classificação

Modelos Lineares

Em aprendizado supervisionado, é comum termos:

- Uma variável independente (ou variável-alvo, ou ainda resultado) Y
- Uma ou mais variáveis dependentes (ou covariantes) $X_1 \ldots X_p$
- Um conjunto de observações, em que Y e X_i são observadas

Difere do não supervisionado em que neste último nem sempre se conhece Y

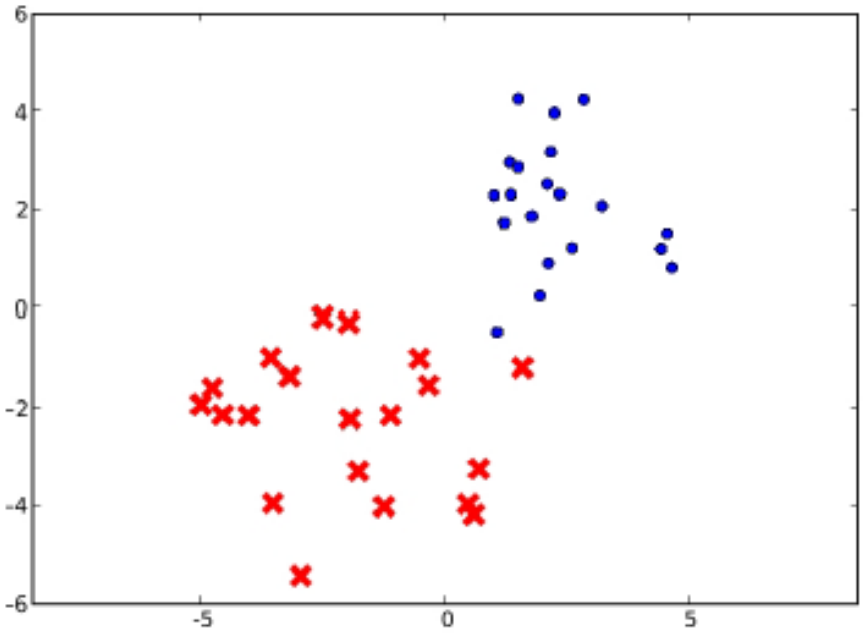
Classificação

Na classificação, além do citado, temos um conjunto pré-definido de classes $C_1 \ldots C_j$ às quais está associado ao valor y da variável Y

Queremos então um procedimento que possa ser aplicado a uma seqüência de casos, de modo a que novas instâncias possam ser associados a uma de um conjunto de classes pré-definidas ($Y = y \in \{C_1 \ldots C_j\}$), com base em seu atributos observados $X_i = x_i$.

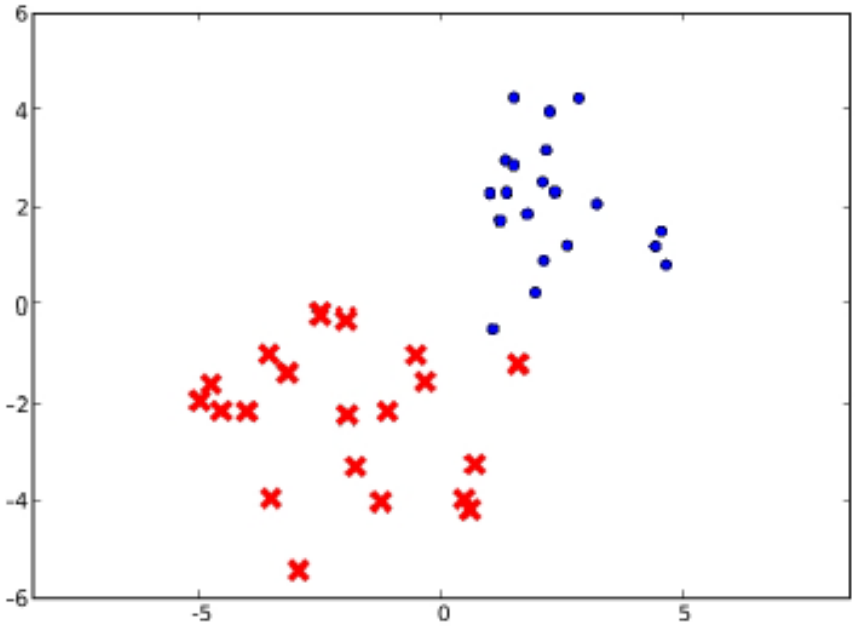
Classificação

- Imagine que queremos separar os Xs e Os (exemplos pré-classificados nessas duas categorias)
  - Cada ponto é descrito por um número de atributos
    - No exemplo, 2: eixo x e y
  - Queremos achar um meio de classificar novos exemplos nessas categorias



Classificação

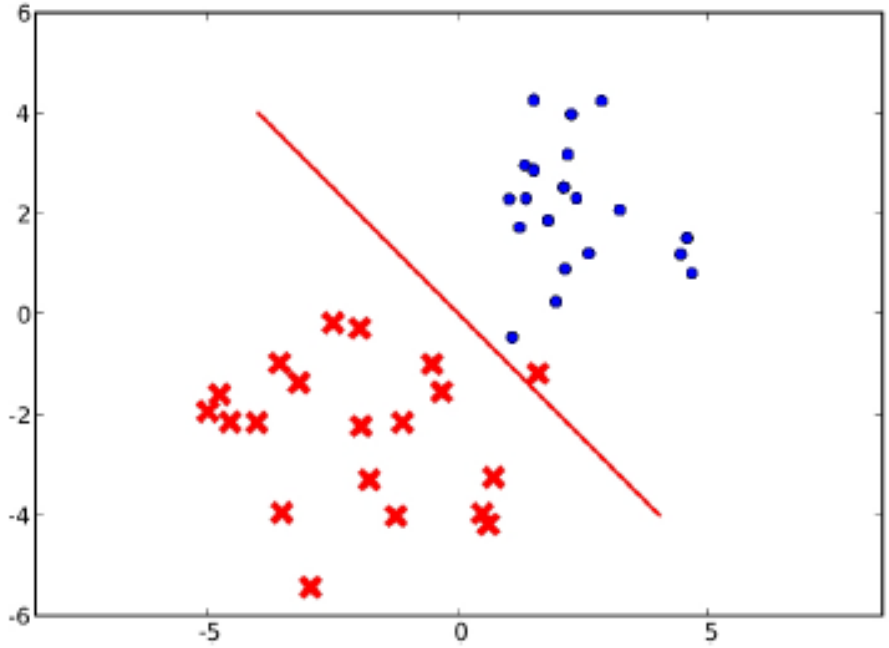
- Imagine que queremos separar os Xs e Os (exemplos pré-classificados nessas duas categorias)
  - Cada ponto é descrito por um número de atributos
    - No exemplo, 2: eixo x e y
  - Queremos achar um meio de classificar novos exemplos nessas categorias



- Como fazer?

Classificação

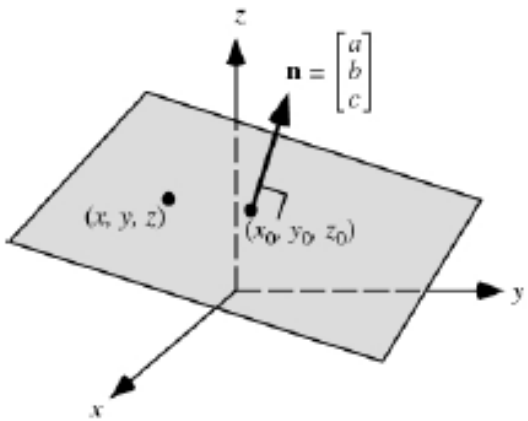
- Modo mais simples: divida os pontos com uma linha – **classificação linear**
  - A classificação é feita com base no valor de uma combinação linear dos atributos
  - Em mais de 2 dimensões, são hiperplanos, não linhas



Classificação Linear

- O hiperplano irá obedecer à seguinte equação:
  $\omega_1 x_1 + \omega_2 x_2 + \ldots \omega_d x_d + \omega_0 = 0$
  - Inclinação: determinada pelo vetor $(\omega_1, \ldots, \omega_d)$ (sua normal)
  - Localização: determinada pelo **viés** ω_0
  - Geometricamente, o vetor de parâmetros é perpendicular ao hiperplano

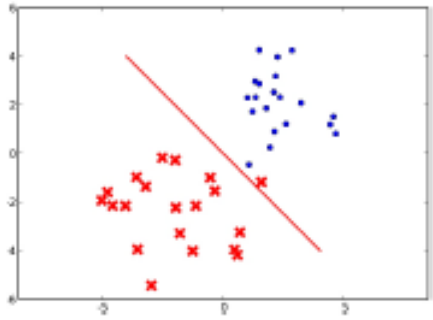
$$\text{plano: } \vec{\omega} \cdot (\vec{x} - \vec{x_0}) = 0$$
$$\Rightarrow |\vec{\omega}||(\vec{x} - \vec{x_0})|cos(\theta) = 0$$
$$\Rightarrow \theta = 90^o$$
$$\text{e } \omega_0 \equiv -\omega_1 x_{01} - \ldots - \omega_d x_{0d}$$



Classificação Linear

- Como classificar?
  - Seja $f(\vec{x}) = \omega_1 x_1 + \omega_2 x_2 + \ldots \omega_d x_d + \omega_0$
  - Então

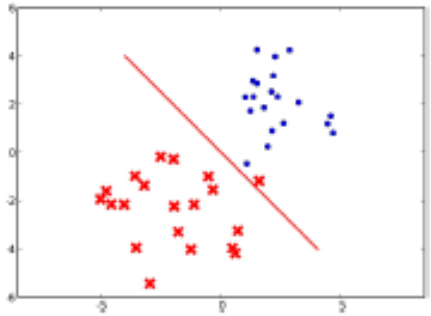
$$h(\vec{x}) = sinal(f(\vec{x})) = \begin{cases} c & \text{se } f(\vec{x}) \geq 0 \\ \bar{c} & \text{se } f(\vec{x}) < 0 \end{cases}$$



Classificação Linear

- Como classificar?
  - Seja $f(\vec{x}) = \omega_1 x_1 + \omega_2 x_2 + \ldots \omega_d x_d + \omega_0$
  - Então

$$h(\vec{x}) = sinal(f(\vec{x})) = \begin{cases} c & \text{se } f(\vec{x}) \geq 0 \\ \bar{c} & \text{se } f(\vec{x}) < 0 \end{cases}$$



- Assim, dado um ponto $p = (x, y)$, e o plano $\omega_1 x + \omega_2 y + \omega_0 = 0$, temos que:

$$h(\vec{p}) = sinal(\omega_1 x + \omega_2 y + \omega_0) = \begin{cases} c & \text{se } f(\vec{p}) \geq 0 \\ \bar{c} & \text{se } f(\vec{p}) < 0 \end{cases}$$

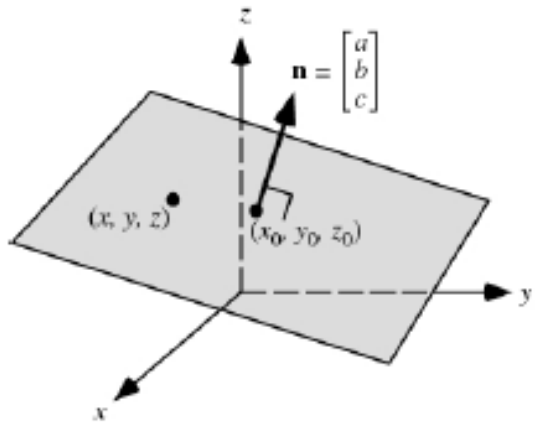
Classificação Linear

- Exemplo:
  - Considere $\vec{x} = (x_1, \ldots, x_d)$ os atributos de um cliente
  - Seu crédito será aprovado se $\sum_{i=1}^{d} \omega_i x_i \geq \mathit{limiar}$
  - Será negado se $\sum_{i=1}^{d} \omega_i x_i < \mathit{limiar}$
- Note que $h(x)$ pode ser reescrita como:

$$h(\vec{x}) = \mathit{sinal}\left(\left(\sum_{i=1}^{d} \omega_i x_i\right) - \mathit{limiar}\right)$$

Classificação Linear

- Para fins práticos, podemos criar uma coordenada artificial $x_0 = 1$, e



$$\omega_1 x_1 + \omega_2 x_2 + \ldots \omega_d x_d + \omega_0$$

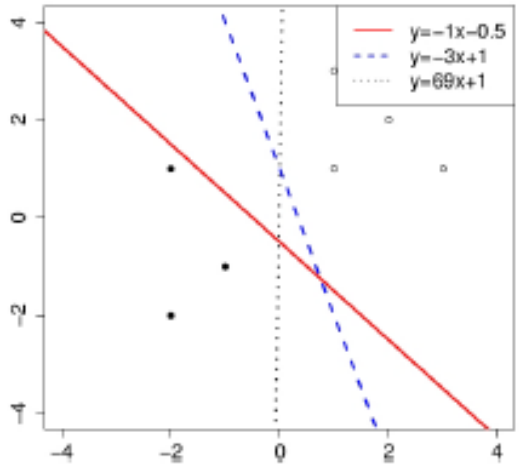
$$\Rightarrow \left(\sum_{i=1}^{d} \omega_i x_i \right) + \omega_0$$

onde $\omega_0 = -limiar$. Com $x_0 = 1$ $h(\vec{x})$ pode ser então simplificada para

$$h(\vec{x}) = sinal \left(\sum_{i=0}^{d} \omega_i x_i \right)$$

Classificação Linear

- E qual o melhor hiperplano?
  - Definido pelo processo de aprendizado – exemplos são usados para se achar "bons" ω_i
  - Diferentes noções de "bom" existem $\rightarrow$ diferentes algoritmos



- Técnicas para classificação linear:
  - Redes Neurais (multi-layer perceptron)
  - Naïve Bayes
  - SVM
  - Regressão Logística

Classificação Linear

- Qualquer que seja o algoritmo, as hipóteses sempre obedecerão a

$$h(\vec{x}) = sinal\left(\left(\sum_{i=1}^{d} \omega_i x_i\right) - limiar\right)$$

  - O que define a hipótese h é então a escolha de ω_i e *limiar*
  - Isso é o que o algoritmo precisa variar para escolher a melhor hipótese

Regressão Linear

Regressão Linear

- Classificação:
  - Tenta classificar um exemplo em um conjunto de classes pré-determinadas
  - Ex:
    - Aprovação de crédito: (sim/não)
- Regressão:
  - Tenta descobrir o valor associado a um novo exemplo
  - A diferença está em que na regressão a variável-alvo é numérica e contínua
  - Ex:
    - Linha de crédito: quantos R$ podem ser emprestados

Regressão Linear

- Considere o problema do crédito:
  - Entrada: $x =$

idade	23
salário anual	R$30.000,00
anos na residência	1
anos no trabalho	1
dívida atual	R$15.000,00
...	...

  - Hipótese: $h(\vec{x}) = \sum_{i=0}^{d} \omega_i x_i = \vec{\omega} \cdot \vec{x}$

Regressão, pela saída real, e linear pela forma de tratamento da entrada ser linear (note a remoção da função *sinal*)

Regressão Linear

- Dados de entrada:

  $(\vec{x_1}, y_1), (\vec{x_2}, y_2), \ldots, (\vec{x_N}, y_N)$
  onde
  $\vec{x_i} = \{x_j\}$ corresponde ao pedido de crédito feito pelo i-ésimo cliente (atributos)
  $y_i \in \mathbb{R}$ ao montante concedido

- A regressão irá tentar replicar as decisões tomadas pelas pessoas
  - A questão é: com que precisão? Ou
  - Quão boa é a aproximação entre $h(\vec{x}) = \vec{\omega} \cdot \vec{x}$ e $f(\vec{x})$ (a função que hipoteticamente determina os dados reais)

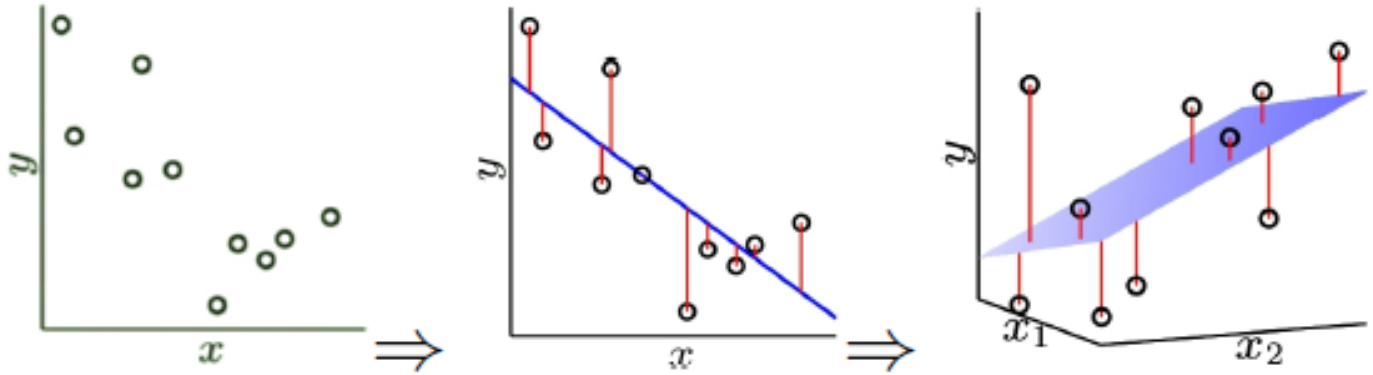
Regressão Linear

- Medindo o erro:
  - Importante para saber qual hipótese $h(\vec{x})$ melhor aproxima a função-alvo $f(\vec{x})$
  - O algoritmo irá tentar minimizar o erro da amostra movendo-se de uma hipótese a outra
  - No caso da regressão linear, usamos o erro quadrático $(h(\vec{x}) - f(\vec{x}))^2$
    - Se usarmos meramente a diferença entre previsto e real, valores positivos podem cancelar negativos
    - Calculamos então a média do erro da amostra para cada ponto (dos N) que temos:

$$E(h) = \frac{1}{N} \sum_{i=1}^{N} (h(\vec{x_i}) - y_i)^2$$

Regressão Linear

- Medindo o erro: Exemplo



Note que, na verdade, x seria x_1, dado que há no modelo um x_0 que incluimos

- A regressão irá tentar produzir uma linha ajustada aos dados de acordo com o erro quadrático
- Em mais de 2 dimensões, produzirá um plano

Regressão Linear - Medindo o erro

- Variação do método dos quadrados mínimos:
  - Com quadrados mínimos minimizamos a soma dos erros quadráticos
  - Aqui minimizamos a média
- Considere a expressão do erro:

$$E(h) = \frac{1}{N}\sum_{i=1}^{N}(h(\vec{x_i}) - y_i)^2$$

como $h(\vec{x}) = \vec{\omega} \cdot \vec{x}$, então temos

$$E(\vec{\omega}) = \frac{1}{N}\sum_{i=1}^{N}(\vec{\omega} \cdot \vec{x_i} - y_i)^2$$

Regressão Linear - Medindo o erro

- e...

$$E(\vec{\omega}) = \frac{1}{N}\sum_{i=1}^{N}(\vec{\omega}\cdot\vec{x_i} - y_i)^2$$

$$= \frac{1}{N}|X\vec{\omega} - \vec{y}|^2$$

onde

$$X = \begin{bmatrix} x_{1,0} & x_{1,1} & \cdots & x_{1,d} \\ x_{2,0} & x_{2,1} & \cdots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,0} & x_{N,1} & \cdots & x_{N,d} \end{bmatrix}$$

Atributos (0–d) de cada exemplo.

$$\vec{\omega} = \begin{bmatrix} \omega_0 \\ \omega_1 \\ \vdots \\ \omega_d \end{bmatrix}$$

Pesos de cada atributo.

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

Saída de cada exemplo.

Regressão Linear - Medindo o erro

- Por partes...

$$X\vec{\omega} = \begin{bmatrix} x_{1,0} & x_{1,1} & \cdots & x_{1,d} \\ x_{2,0} & x_{2,1} & \cdots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,0} & x_{N,1} & \cdots & x_{N,d} \end{bmatrix} \begin{bmatrix} \omega_0 \\ \omega_1 \\ \vdots \\ \omega_d \end{bmatrix} = \begin{bmatrix} x_{1,0}\omega_0 + x_{1,1}\omega_1 + \cdots + x_{1,d}\omega_d \\ x_{2,0}\omega_0 + x_{2,1}\omega_1 + \cdots + x_{2,d}\omega_d \\ \ddots \\ x_{N,0}\omega_0 + x_{N,1}\omega_1 + \cdots + x_{N,d}\omega_d \end{bmatrix}$$

$$X\vec{\omega} - \vec{y} = X\vec{\omega} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} x_{1,0}\omega_0 + x_{1,1}\omega_1 + \cdots + x_{1,d}\omega_d - y_1 \\ x_{2,0}\omega_0 + x_{2,1}\omega_1 + \cdots + x_{2,d}\omega_d - y_2 \\ \ddots \\ x_{N,0}\omega_0 + x_{N,1}\omega_1 + \cdots + x_{N,d}\omega_d - y_N \end{bmatrix}$$

Regressão Linear - Medindo o erro

- Reescrevendo...

$$X\vec{\omega} - \vec{y} = \begin{bmatrix} x_{1,0}\omega_0 + x_{1,1}\omega_1 + \cdots + x_{1,d}\omega_d - y_1 \\ x_{2,0}\omega_0 + x_{2,1}\omega_1 + \cdots + x_{2,d}\omega_d - y_2 \\ \ddots \\ x_{N,0}\omega_0 + x_{N,1}\omega_1 + \cdots + x_{N,d}\omega_d - y_N \end{bmatrix} = \begin{bmatrix} \sum_{j=0}^{d}(x_{1,j}\omega_j) - y_1 \\ \sum_{j=0}^{d}(x_{2,j}\omega_j) - y_2 \\ \ddots \\ \sum_{j=0}^{d}(x_{N,j}\omega_j) - y_N \end{bmatrix}$$

Regressão Linear - Medindo o erro

$$|X\vec{\omega} - \vec{y}|^2 = \left(\sum_{j=0}^{d}(x_{1,j}\omega_j) - y_1\right)^2 + \cdots + \left(\sum_{j=0}^{d}(x_{N,j}\omega_j) - y_N\right)^2$$

$$\Rightarrow |X\vec{\omega} - \vec{y}|^2 = \sum_{i=1}^{N}\left(\sum_{j=0}^{d}(x_{i,j}\omega_j) - y_i\right)^2$$

$$E(\vec{\omega}) = \frac{1}{N}|X\vec{\omega} - \vec{y}|^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=0}^{d}(x_{i,j}\omega_j) - y_i\right)^2$$

$$\text{como } h(\vec{x_i}) = \sum_{j=0}^{d}\omega_j x_{i,j} \Rightarrow E(\vec{\omega}) = \frac{1}{N}\sum_{i=1}^{N}(h(\vec{x_i}) - y_i)^2$$

Regressão Linear - Minimizando o Erro

- Para minimizar $E(\vec{\omega})$, tomamos $\nabla E(\vec{\omega}) = \vec{0}$
  - Recorde que $\nabla f = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n} \right)$
- A questão é: minimizar em relação a que?

Regressão Linear - Minimizando o Erro

- Para minimizar $E(\vec{\omega})$, tomamos $\nabla E(\vec{\omega}) = \vec{0}$
  - Recorde que $\nabla f = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n} \right)$
- A questão é: minimizar em relação a que?
  - Não queremos o ponto $(x_0, \ldots, x_d)$ (ou seja, atributos) em que o erro é mínimo, mas sim o conjunto de pesos $(\omega_0, \ldots, \omega_d)$ que minimiza o erro

Regressão Linear - Minimizando o Erro

- Para minimizar $E(\vec{\omega})$, tomamos $\nabla E(\vec{\omega}) = \vec{0}$
  - Recorde que $\nabla f = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n} \right)$
- A questão é: minimizar em relação a que?
  - Não queremos o ponto $(x_0, \ldots, x_d)$ (ou seja, atributos) em que o erro é mínimo, mas sim o conjunto de pesos $(\omega_0, \ldots, \omega_d)$ que minimiza o erro
  - Então...

$$\nabla E(\vec{\omega}) = \left(\frac{\partial E(\vec{\omega})}{\partial \omega_0}, \frac{\partial E(\vec{\omega})}{\partial \omega_1}, \ldots, \frac{\partial E(\vec{\omega})}{\partial \omega_d} \right)$$

Regressão Linear - Minimizando o Erro

- Como $E(\vec{\omega}) = \frac{1}{N}|X\vec{\omega} - \vec{y}|^2$, X e $\vec{y}$ tornam-se constantes
  - São os dados (dados de entrada e saída) que alguém nos deu
  - O parâmetro a ser manipulados é ω
- Então...

$$\nabla E(\vec{\omega}) = \frac{2}{N}X^t(X\vec{\omega} - \vec{y})$$

Regressão Linear - Minimizando o Erro

- Novamente...

$$E(\vec{\omega}) = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=0}^{d}(x_{i,j}\omega_j) - y_i\right)^2$$

$$\Rightarrow \nabla E(\vec{\omega}) = \begin{bmatrix} \frac{\partial E(\vec{\omega})}{\partial \omega_0} \\ \vdots \\ \frac{\partial E(\vec{\omega})}{\partial \omega_d} \end{bmatrix} = \begin{bmatrix} \frac{2}{N}\sum_{i=1}^{N}\left[x_{i,0}\left(\sum_{j=0}^{d}(x_{i,j}\omega_j) - y_i\right)\right] \\ \vdots \\ \frac{2}{N}\sum_{i=1}^{N}\left[x_{i,d}\left(\sum_{j=0}^{d}(x_{i,j}\omega_j) - y_i\right)\right] \end{bmatrix}$$

Regressão Linear - Minimizando o Erro

- Olhando para $X^t(X\vec{\omega} - \vec{y})$ (slide 22):

$$X\vec{\omega} - \vec{y} = \begin{bmatrix} x_{1,0}\omega_0 + x_{1,1}\omega_1 + \cdots + x_{1,d}\omega_d - y_1 \\ x_{2,0}\omega_0 + x_{2,1}\omega_1 + \cdots + x_{2,d}\omega_d - y_2 \\ \vdots \\ x_{N,0}\omega_0 + x_{N,1}\omega_1 + \cdots + x_{N,d}\omega_d - y_N \end{bmatrix}$$

$$\Rightarrow X\vec{\omega} - \vec{y} = \begin{bmatrix} \sum_{j=0}^{d} x_{1,j}\omega_j - y_1 \\ \vdots \\ \sum_{j=0}^{d} x_{N,j}\omega_j - y_N \end{bmatrix}$$

Regressão Linear - Minimizando o Erro

- Então:

$$X^t(X\vec{\omega} - \vec{y}) = \begin{bmatrix} x_{1,0} & x_{2,0} & \cdots & x_{N,0} \\ x_{1,1} & x_{2,1} & \cdots & x_{N,1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1,d} & x_{2,d} & \cdots & x_{N,d} \end{bmatrix} \begin{bmatrix} \sum_{j=0}^{d} x_{1,j}\omega_j - y_1 \\ \sum_{j=0}^{d} x_{2,j}\omega_j - y_2 \\ \vdots \\ \sum_{j=0}^{d} x_{N,j}\omega_j - y_N \end{bmatrix}$$

Regressão Linear - Minimizando o Erro

- $\frac{2}{N} X^t (X\vec{\omega} - \vec{y}) =$

$$\begin{bmatrix} \frac{2}{N}\left[x_{1,0}\left(\sum_{j=0}^{d} x_{1,j}\omega_j - y_1\right)\right] + & \cdots & +\frac{2}{N}\left[x_{N,0}\left(\sum_{j=0}^{d} x_{N,j}\omega_j - y_N\right)\right] \\ \vdots & \ddots & \vdots \\ \frac{2}{N}\left[x_{1,d}\left(\sum_{j=0}^{d} x_{1,j}\omega_j - y_1\right)\right] + & \cdots & +\frac{2}{N}\left[x_{N,d}\left(\sum_{j=0}^{d} x_{N,j}\omega_j - y_N\right)\right] \end{bmatrix}$$

Que corresponde a $\nabla E(\vec{\omega})$, visto no slide 27

Regressão Linear - Minimizando o Erro

- Então, para minimizar o erro, temos que

$$\nabla E(\vec{\omega}) = \frac{2}{N} X^t (X\vec{\omega} - \vec{y}) = \vec{0}$$

$$\Rightarrow X^t (X\vec{\omega} - \vec{y}) = \vec{0}$$

$$\Rightarrow X^t X\vec{\omega} = X^t \vec{y}$$

e $\vec{\omega} = (X^t X)^{-1} X^t \vec{y} = X^\dagger \vec{y}$, onde $X^\dagger = (X^t X)^{-1} X^t$

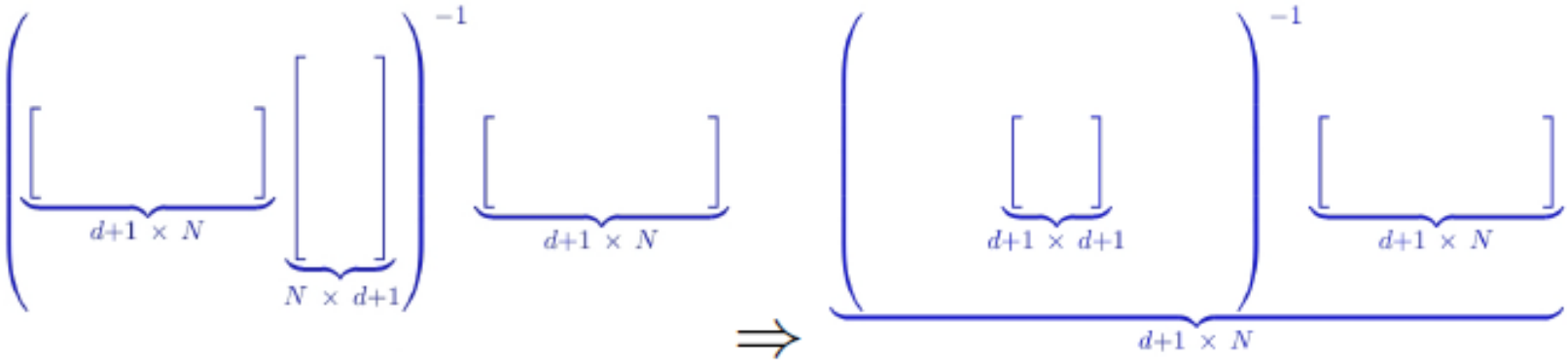
$X^\dagger$ é a pseudo-inversa de X

Regressão Linear - Minimizando o Erro

- Vale lembrar que X^tX não necessariamente é inversível
  - X^tX é quadrada, o que ajuda
  - Para ser inversível, $A^{-1}A = I$ (matriz identidade)
- Se X fosse quadrada e inversível, poderíamos fazer $X\vec{\omega} - \vec{y} = 0$
  - Bastaria multiplicar ambos os lados por X^{-1} e teríamos $\vec{\omega} = X^{-1}\vec{y}$
  - Mas X ser quadrada implicaria termos o mesmo número de exemplos e atributos
    - Normalmente temos poucos atributos e muitos exemplos

Regressão Linear - Minimizando o Erro

- Ainda assim, $X^{\dagger}$ é computacionalmente interessante:



Como em geral $N >> d$, isso torna-se bem atraente

Regressão Linear - Algoritmo

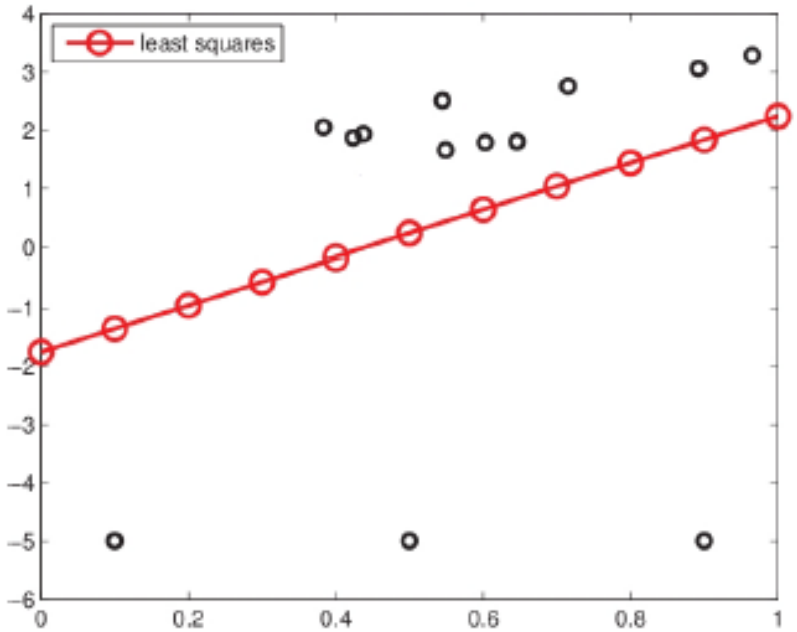
- Construa a matriz X e o vetor $\vec{y}$ a partir do conjunto de dados $(x_1, y_1), \ldots, (x_N, y_N)$ como segue:

$$\underbrace{X = \begin{bmatrix} x_{1,0} & x_{1,1} & \cdots & x_{1,d} \\ x_{2,0} & x_{2,1} & \cdots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,0} & x_{N,1} & \cdots & x_{N,d} \end{bmatrix}}_{\text{matriz de dados de entrada}}, \quad \underbrace{\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{vetor-alvo}}$$

- Calcule a pseudo-inversa $X^\dagger = (X^t X)^{-1} X^t$
- Retorne $\vec{\omega} = X^\dagger \vec{y}$

Regressão Linear - Limitações

- Se houver anomalias (outliers) nos dados, isso pode resultar em um ajuste ruim:



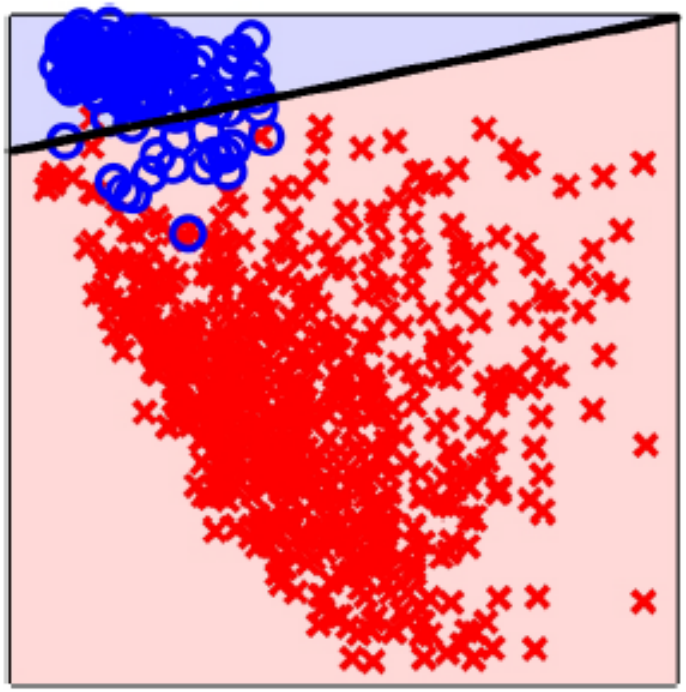
  - Isso se deve ao fato do erro penalizar desvios quadraticamente $\rightarrow$ pontos longe da reta a afetam mais que pontos próximos

Regressão Linear - Balanço Final

- Prós:
  - Resultados de fácil interpretação
  - Computacionalmente barato
  - Trabalha tanto com valores numéricos quanto nominais
    - Basta mapear as categorias a valores. Ex: $c_1 = +1$ e $c_2 = -1$ (note que $\pm 1 \in \mathbb{R}$)
    - Use a regressão linear para obter $\vec{\omega}$, onde $\vec{\omega} \cdot \vec{x}_i \approx y_i = \pm 1$
    - Estando próximo de ± 1, $sinal(\vec{\omega} \cdot \vec{x})$ estará correto
- Contras:
  - Modela de maneira pobre dados não lineares

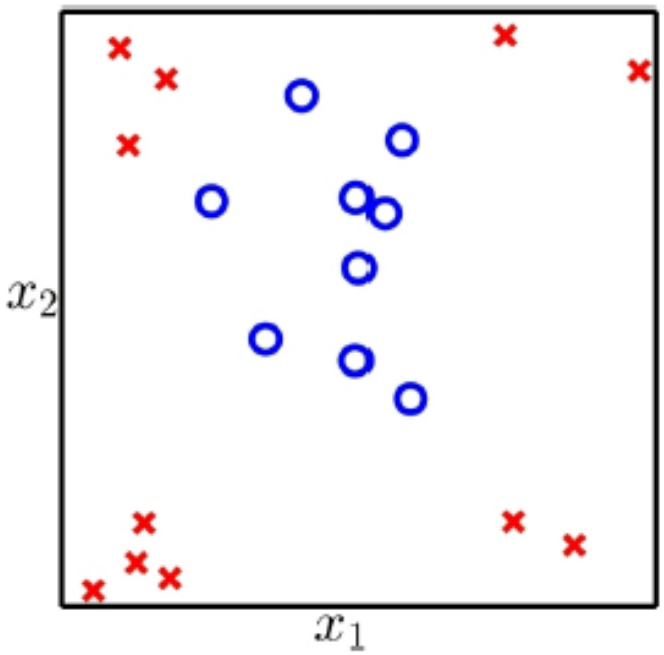
Regressão Linear para Classificação

- Problemas no uso como classificação:
  - Tentará fazer todos os pontos ± 1 e não conseguirá
  - Considera -2, -3 etc como erros, quando estariam corretos na classificação
  - Vai minimizar o erro
  - Ainda assim, boa aproximação inicial



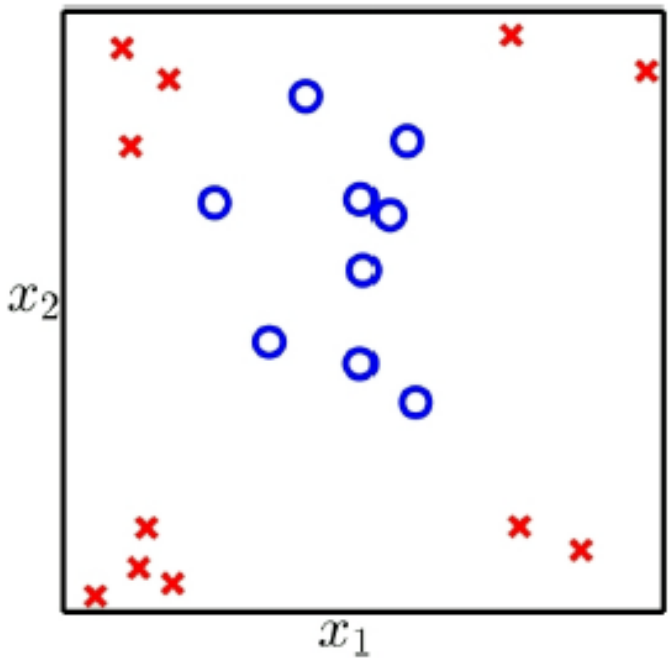
Transformação Não-Linear

- Considere o seguinte conjunto de dados
  - São linearmente separáveis?



Transformação Não-Linear

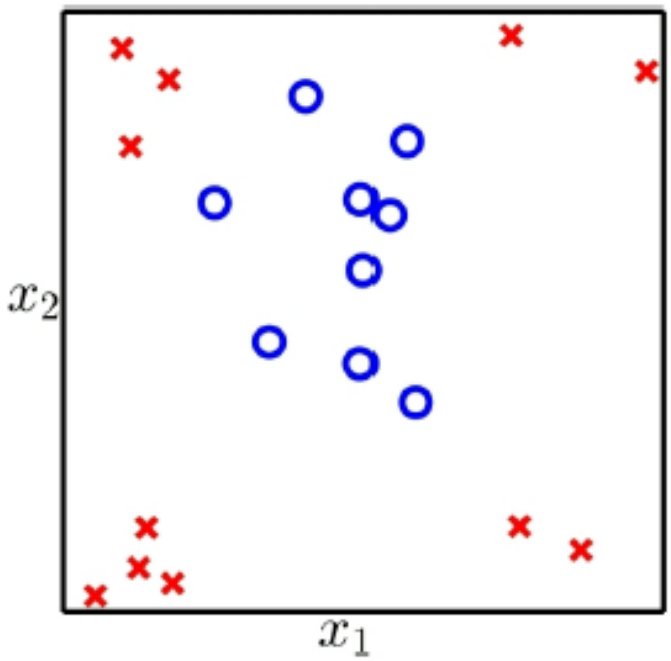
- Considere o seguinte conjunto de dados
  - São linearmente separáveis?
  - Linearmente em que?



Transformação Não-Linear

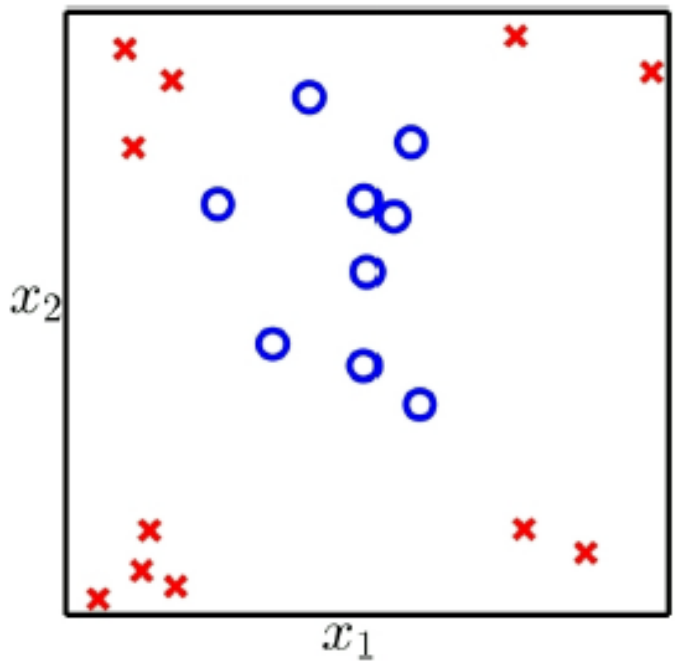
- Considere o seguinte conjunto de dados
  - São linearmente separáveis?
  - Linearmente em que?
- A regressão implementa

$$\sum_{i=0}^{d} \omega_i x_i$$



Transformação Não-Linear

- Considere o seguinte conjunto de dados
  - São linearmente separáveis?
  - Linearmente em que?



- A regressão implementa

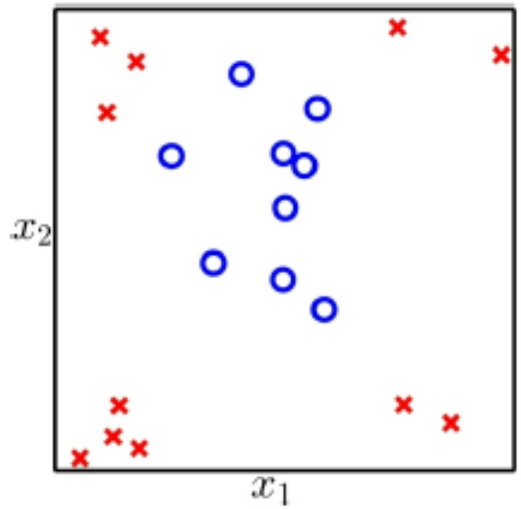
$$\sum_{i=0}^{d} \omega_i x_i$$

  - Basta tomarmos os dados como lineares em relação aos pesos, não a x_i

Transformação Não-Linear

- Como?
  - Repare que se colocarmos a origem no centro do gráfico, os azuis estarão mais perto dela e os vermelhos mais longe
  - Basta então dar mais peso à distância ao centro, mudando o sistema de coordenadas:

$$(x_1, x_2) \xrightarrow{\Phi} (x_1^2, x_2^2)$$

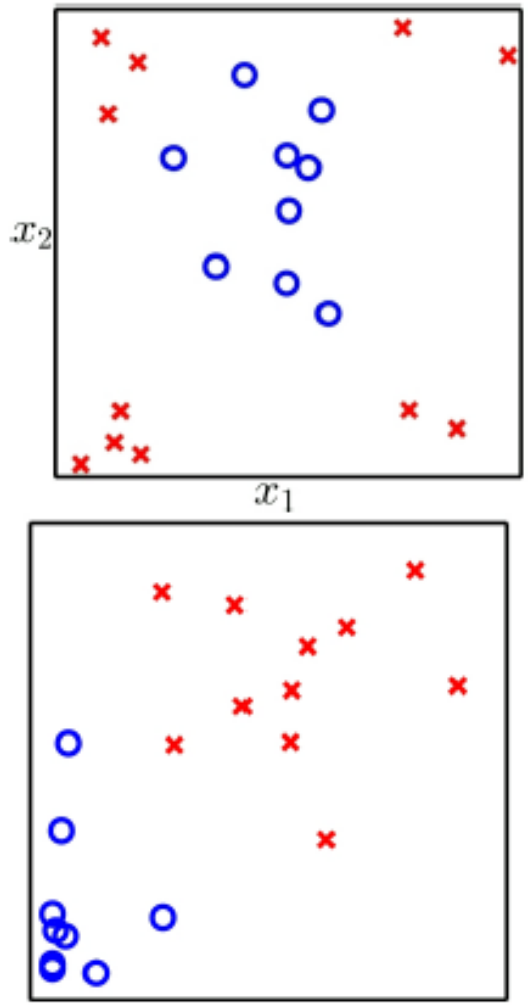


Transformação Não-Linear

- Como?
  - Repare que se colocarmos a origem no centro do gráfico, os azuis estarão mais perto dela e os vermelhos mais longe
  - Basta então dar mais peso à distância ao centro, mudando o sistema de coordenadas:

$$(x_1, x_2) \xrightarrow{\Phi} (x_1^2, x_2^2)$$

  - E o novo gráfico, com essa transformação, fica:

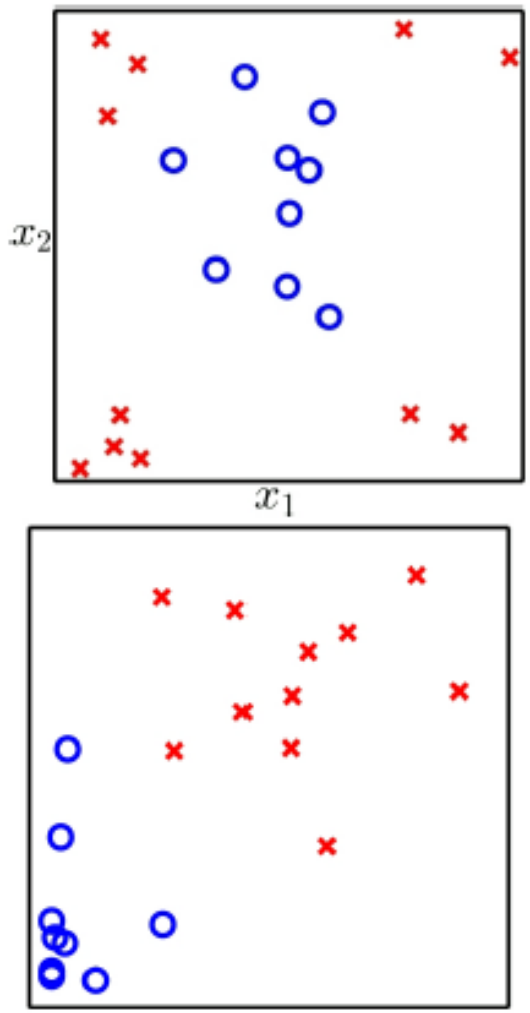


Transformação Não-Linear

- Como?
  - Repare que se colocarmos a origem no centro do gráfico, os azuis estarão mais perto dela e os vermelhos mais longe
  - Basta então dar mais peso à distância ao centro, mudando o sistema de coordenadas:

$$(x_1, x_2) \xrightarrow{\Phi} (x_1^2, x_2^2)$$

  - E o novo gráfico, com essa transformação, fica:
    - Linearmente separáveis



Regressão- Passos

- Coleta dos dados
- Preparação
  - Se numéricos, ok, se nominais, mapeie-os a valores numéricos
- Treinamento
  - Encontre os pesos da regressão
- Teste
  - Meça a correlação entre os valores previstos e os dados reais, para medir o sucesso do modelo

Quadrados Mínimos Regularizado

- A fim de controlar o sobre-ajuste, a função erro a ser minimizada pode ser definida como

$$E(\vec{\omega}) + \gamma E_{\vec{\omega}}(\vec{\omega})$$

onde γ é coeficiente de regularização que controla a importância relativa do erro dependente dos dados e o termo de regularização $E_{\vec{\omega}}(\vec{\omega})$.

- Uma das formas mais simples para termo de regularização é dada pela equação abaixo

$$E_{\vec{\omega}}(\vec{\omega}) = \vec{\omega}^t\vec{\omega}$$

- Considerando a função erro como a soma dos quadrados do erro dada por

$$E(\vec{\omega}) = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=0}^{d}(x_{i,j}\omega_j) - y_i\right)^2$$

Quadrados Mínimos Regularizado

- A função erro total é definida como

$$\frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=0}^{d}(x_{i,j}\omega_j) - y_i\right)^2 + \gamma\vec{\omega}^t\vec{\omega}$$

- Esta escolha particular para o termo de regularização é conhecida na literatura de Aprendizado de Máquina como decaimento dos pesos por que em algoritmos de aprendizado sequencial, este encoraja os valores dos pesos cair em direção a zero.
- Em estatistica, este fornece um exemplo de um método de encolhimento de parâmetro, por que este reduz os valores dos paramentros em direção a zero

Quadrados Mínimos Regularizado

- Tomando o gradiente com relação a w, fazendo igual a zero e isolando w, nós obtemos

$$\vec{\omega} = \left(\frac{\gamma}{N} I + X^t X\right)^{-1} X^t \vec{y}$$

- Este representa uma extensão da solução do Quadrados Mínimos
- Um termo de regularização mais geral é algumas vezes usada, para o qual o erro de regularizado assume a seguinte forma

$$\frac{1}{N} \sum_{i=1}^{N} \left(\sum_{j=0}^{d} (x_{i,j} \omega_j) - y_i \right)^2 + \gamma \sum_{j=1}^{M} |w_j|^q$$

- $q = 2$ corresponde para um regularizador quadrático
- $q = 1$ é conhecido como *lasso* na literatura de estatistica.

Regressão Logistica

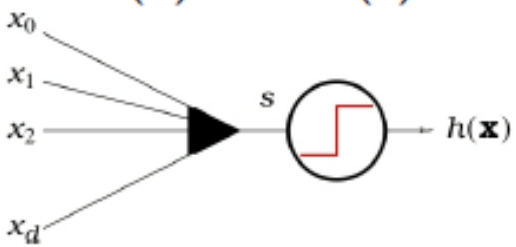
Regressão Logistica

No modelo linear combinamos linearmente a entrada com os pesos:

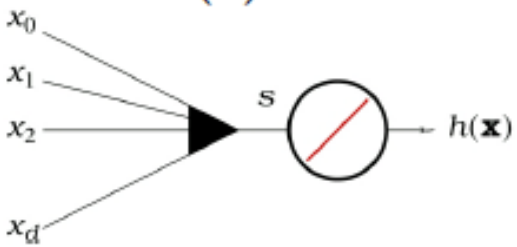
$$s = \sum_{i=0}^{d} \omega_i x_i$$

E então fazemos algo com o resultado, que pode ser:

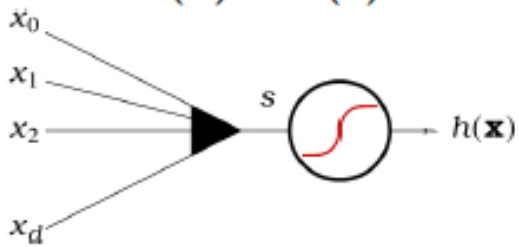
Classificação linear
$h(\vec{x}) = sinal(s)$



Regressão linear
$h(\vec{x}) = s$



Regressão logística
$h(\vec{x}) = \theta(s)$

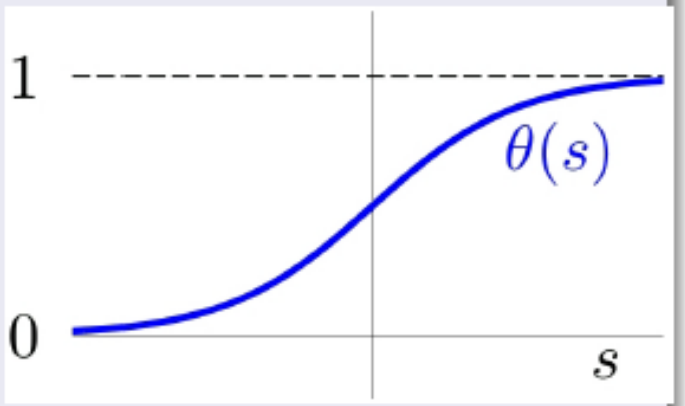


Regressão Logistica

A Função Logística

Pegamos s e aplicamos uma não-linearidade θ a ele $\rightarrow$ a Função Logística:

$$\theta(s) = \frac{e^s}{1 + e^s}$$

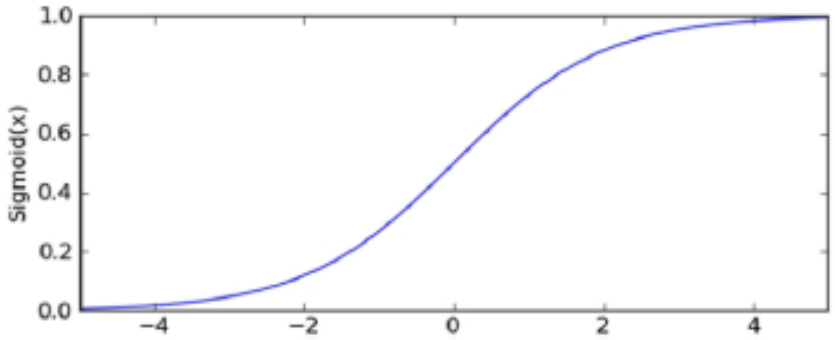


Em regressão logística a saída $h(\vec{x}) = \theta(s)$ é interpretada como uma probabilidade

- Varia de 0 a 1. Se s muito negativo, $\theta(s)$ aproxima-se de 0, se muito positivo, de 1
- Há um *limiar suave* (soft threshold)

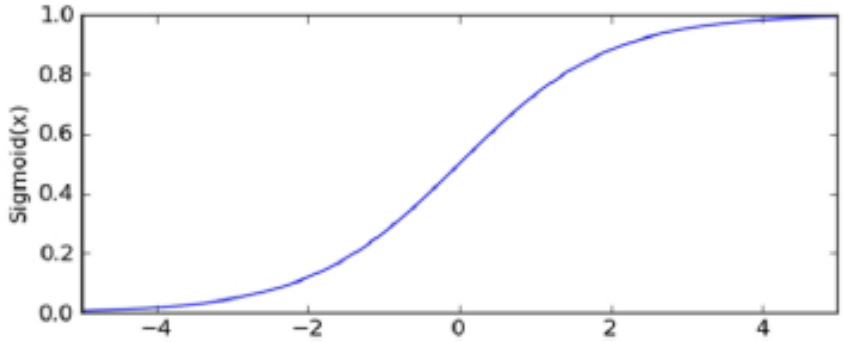
Regressão Logistica

Ainda que a função logística pareça distante da função degrau usada na classificação . . .

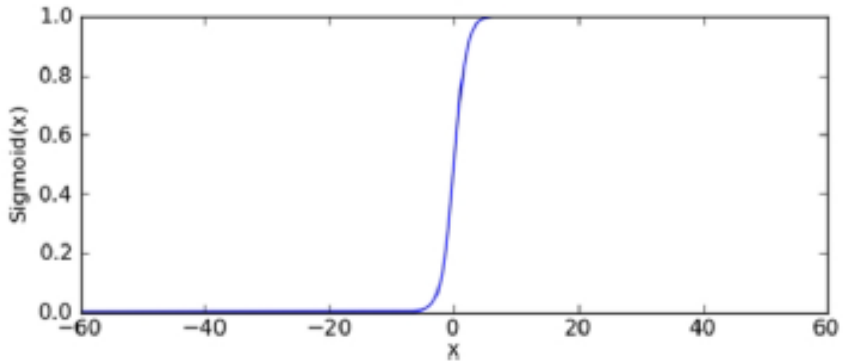


Regressão Logistica

Ainda que a função logística pareça distante da função degrau usada na classificação ...



Em uma escala grande o suficiente, a sigmóide parece com a função degrau



RLog - Interpretação Logistica

Exemplo

Previsão de ataques cardíacos em dado intervalo de tempo:

- Entrada $\vec{x}$: nível de colesterol, idade, peso etc
- Saída $\theta(s)$: probabilidade de um ataque cardíaco
- Lembrando que o sinal $s = \vec{\omega} \cdot \vec{x} \rightarrow$ fator de risco
  - Note que s permanece linear...

Modelo mais interessante (e realista) que uma simples classificação, em que a resposta seria que terá (ou não) o ataque $\rightarrow$ 100% de certeza

RLog - Interpretação Logistica

Suponha que temos um conjunto de dados $(\vec{x}, y)$, com y binário, gerado por uma função-alvo com ruído:

$$P(y|\vec{x}) = \begin{cases} f(\vec{x}) & \text{para } y = +1 \\ 1 - f(\vec{x}) & \text{para } y = -1 \end{cases}$$

Junto a cada ponto $(\vec{x})$, é dada uma uma saída (y) que é afetada por uma probabilidade

- Probabilidade de ataque cardíaco $(y = +1)$ ou não $(y = -1)$, dado os dados

RLog - Interpretação Logistica

$$P(y|\vec{x}) = \begin{cases} f(\vec{x}) & \text{para } y = +1 \\ 1 - f(\vec{x}) & \text{para } y = -1 \end{cases}$$

Estamos tentando aprender a função-alvo $f : \mathbb{R}^d \rightarrow [0, 1]$

- A probabilidade de um ataque

Queremos então aprender $g(\vec{x}) = \theta(\vec{\omega} \cdot \vec{x}) \approx f(\vec{x})$

- A hipótese final que aproxima f

A questão é então como escolho os pesos (parâmetros) de modo a que a hipótese da regressão logística reflita a função-alvo

Regressão Logistica - Medida de Erro

Nesse modelo, para cada $(\vec{x}, y)$, y é gerado pela probabilidade $f(\vec{x})$

- Avaliamos diferentes hipóteses conforme a probabilidade de cada uma delas ser realmente a função-alvo que gerou os dados

Assumamos que uma determinada hipótese represente a função-alvo (ou seja, $h = f$)

- Qual a probabilidade de gerar esses dados se essa suposição for verdadeira?
- Ou, se $h = f$, qual a probabilidade de obtermos y a partir de $\vec{x}$?
- Note que há uma inversão: não queremos saber a probabilidade da hipótese dados os dados, mas sim a probabilidade dos dados dada a hipótese

Regressão Logistica - Medida de Erro

Olhemos então a distribuição de probabilidade que assumimos:

$$P(y|\vec{x}) = \begin{cases} f(\vec{x}) & \text{para } y = +1 \\ 1 - f(\vec{x}) & \text{para } y = -1 \end{cases}$$

Sob a suposição de que $h = f$, essa probabilidade torna-se

$$P(y|\vec{x}) = \begin{cases} h(\vec{x}) & \text{para } y = +1 \\ 1 - h(\vec{x}) & \text{para } y = -1 \end{cases}$$

Regressão Logistica - Medida de Erro

$$P(y|\vec{x}) = \begin{cases} h(\vec{x}) & \text{para } y = +1 \\ 1 - h(\vec{x}) & \text{para } y = -1 \end{cases}$$

Alguns pontos são interessantes:

- $P(y|\vec{x})$ trata de somente um ponto $(\vec{x})$
- Não é interessante que a fórmula trate de casos (descontinuidade) – queremos algo mais analítico.

Note, contudo, que, como $\theta(s) = \dfrac{e^s}{1+e^s}$, então

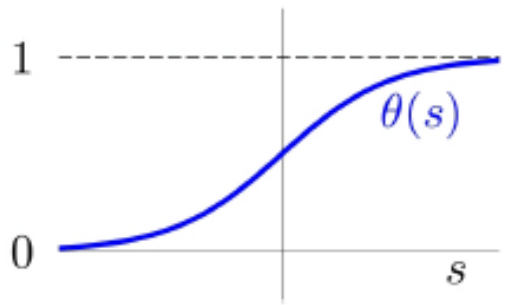
$$\theta(-s) = \frac{e^{-s}}{1+e^{-s}} = \frac{1}{1+e^s} = 1 - \frac{e^s}{1+e^s} = 1 - \theta(s)$$

Regressão Logistica - Medida de Erro

Assim, como $h(\vec{x}) = \theta(s)$, com $s = \vec{\omega} \cdot \vec{x}$

$$P(y|\vec{x}) = \begin{cases} \theta(s) & \text{para } y = +1 \\ \theta(-s) & \text{para } y = -1 \end{cases}$$

(observável no gráfico)



Note então que $P(y|\vec{x}) = \theta(y\vec{\omega} \cdot \vec{x})$:

- Como $y = \pm 1$, isso cobre os dois casos de $\theta(s)$ e $\theta(-s)$
- E, para o conjunto de dados $\mathcal{D} = (\vec{x}_1, y_1), \ldots, (\vec{x}_N, y_N)$, a probabilidade de, sob a hipótese, observarmos cada saída, dado seu respectivo ponto, é:

$$P(\mathcal{D}|h(\vec{x}) = f(\vec{x})) = \prod_{n=1}^{N} P(y_n|\vec{x}_n) = \prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)$$

(assumindo-se independência entre os exemplos fornecidos)

Regressão Logistica - Medida de Erro

$$P(\mathcal{D}|h(\vec{x}) = f(\vec{x})) = \prod_{n=1}^{N} P(y_n|\vec{x}_n) = \prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)$$

Note que o mesmo $\vec{\omega}$ que influencia em um ponto é o que influencia em todos

- O superajuste a um pode prejudicar o ajuste a outro

Queremos então encontrar $\vec{\omega}$ que maximiza essa probabilidade

- $\vec{\omega}$ que minimiza a medida do erro

Regressão Logistica - Medida de Erro

Então, para maximizarmos

$$P(\mathcal{D}|h(\vec{x}) = f(\vec{x})) = \prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)$$

queremos o conjunto de $\vec{\omega}$ que minimize o erro

$$E(\vec{\omega}) = \frac{1}{N} \sum_{n=1}^{N} ln(1 + e^{-y_n \vec{\omega} \cdot \vec{x_n}})$$

O chamado "cross-entropy error"

Regressão Logistica - Maximizando P

Queremos maximizar

$$P(\mathcal{D}|h(\vec{x}) = f(\vec{x})) = \prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)$$

Em relação a que?

Regressão Logistica - Maximizando P

Queremos maximizar

$$P(\mathcal{D}|h(\vec{x}) = f(\vec{x})) = \prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)$$

Em relação a que? A $\vec{\omega}$ (todo o resto é constante)

Regressão Logistica - Maximizando P

Queremos maximizar

$$P(\mathcal{D}|h(\vec{x}) = f(\vec{x})) = \prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)$$

Em relação a que? A $\vec{\omega}$ (todo o resto é constante)

Em vez disso, não poderíamos maximizar?

$$ln(P(\mathcal{D}|h(\vec{x}) = f(\vec{x}))) = ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)$$

Regressão Logistica - Maximizando P

Queremos maximizar

$$P(\mathcal{D}|h(\vec{x}) = f(\vec{x})) = \prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)$$

Em relação a que? A $\vec{\omega}$ (todo o resto é constante)

Em vez disso, não poderíamos maximizar?

$$ln(P(\mathcal{D}|h(\vec{x}) = f(\vec{x}))) = ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)$$

Certamente, pois $P(\mathcal{D}|h(\vec{x}) = f(\vec{x}))$ não será negativa nem zero, e os mesmos parâmetros $\vec{\omega}$ que maximizam uma maximizam a outra

Regressão Logistica - Maximizando P

Ou seja:

$$max(P(\mathcal{D}|h(\vec{x}) = f(\vec{x}))) \iff max\left(ln\left(\prod_{n=1}^{N}\theta(y_n\vec{\omega}\cdot\vec{x}_n)\right)\right)$$

E também nada nos impede de maximizar

$$\frac{1}{N}ln\left(\prod_{n=1}^{N}\theta(y_n\vec{\omega}\cdot\vec{x}_n)\right)$$

Note que estamos tentando chegar a uma expressão do erro...

Regressão Logistica - Maximizando P

Então, temos que:

$$max(P(\mathcal{D}|h(\vec{x}) = f(\vec{x}))) \Rightarrow max\left(\frac{1}{N} ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)\right)$$

E podemos maximizar?

$$-\frac{1}{N} ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)$$

Regressão Logistica - Maximizando P

Então, temos que:

$$max(P(\mathcal{D}|h(\vec{x}) = f(\vec{x}))) \Rightarrow max\left(\frac{1}{N} ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)\right)$$

E podemos maximizar?

$$-\frac{1}{N} ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)$$

Não. Mas se minimizarmos essa expressão, teremos maximizado sua correspondente positiva

Regressão Logistica - Maximizando P

Ou seja:

$$max(P(\mathcal{D}|h(\vec{x}) = f(\vec{x}))) \iff min\left(-\frac{1}{N} ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)\right)$$

Como $ln(a^b) = b \times ln(a)$, então temos que

$$-\frac{1}{N} ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right) = \frac{1}{N} ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)^{-1}$$

E como $(a \times b)^{-1} = a^{-1} \times b^{-1}$, então

$$\frac{1}{N} ln\left(\prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)\right)^{-1} = \frac{1}{N} ln\left(\prod_{n=1}^{N} \frac{1}{\theta(y_n \vec{\omega} \cdot \vec{x}_n)}\right)$$

Regressão Logistica - Maximizando P

E como $ln(a \times b) = ln(a) + ln(b)$, então

$$\frac{1}{N} ln\left(\prod_{n=1}^{N} \frac{1}{\theta(y_n \vec{\omega} \cdot \vec{x}_n)}\right) = \frac{1}{N} \sum_{n=1}^{N} ln\left(\frac{1}{\theta(y_n \vec{\omega} \cdot \vec{x}_n)}\right)$$

Com $\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$, temos que

$$\frac{1}{N} \sum_{n=1}^{N} ln\left(\frac{1}{\theta(y_n \vec{\omega} \cdot \vec{x}_n)}\right) = \frac{1}{N} \sum_{n=1}^{N} ln(1 + e^{-y_n \vec{\omega} \cdot \vec{x}_n}) = E(\vec{\omega})$$

Regressão Logistica - Maximizando P

E como $ln(a \times b) = ln(a) + ln(b)$, então

$$\frac{1}{N} ln\left(\prod_{n=1}^{N} \frac{1}{\theta(y_n \vec{\omega} \cdot \vec{x}_n)}\right) = \frac{1}{N} \sum_{n=1}^{N} ln\left(\frac{1}{\theta(y_n \vec{\omega} \cdot \vec{x}_n)}\right)$$

Com $\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$, temos que

$$\frac{1}{N} \sum_{n=1}^{N} ln\left(\frac{1}{\theta(y_n \vec{\omega} \cdot \vec{x}_n)}\right) = \frac{1}{N} \sum_{n=1}^{N} ln(1 + e^{-y_n \vec{\omega} \cdot \vec{x}_n}) = E(\vec{\omega})$$

Regressão Logistica - Medida do Erro

E, portanto, ao maximizarmos

$$P(\mathcal{D}|h(\vec{x}) = f(\vec{x})) = \prod_{n=1}^{N} \theta(y_n \vec{\omega} \cdot \vec{x}_n)$$

estaríamos minimizando o erro

$$E(\vec{\omega}) = \frac{1}{N} \sum_{n=1}^{N} \underbrace{ln(1 + e^{-y_n \vec{\omega} \cdot \vec{x}_n})}_{erro(h(\vec{x}_n), y_n)}$$

Regressão Logistica - Medida do Erro

Em nosso exemplo anterior...

$$E(\vec{\omega}) = \frac{1}{N}\sum_{n=1}^{N} \underbrace{ln(1 + e^{-y_n \vec{\omega}\cdot\vec{x_n}})}_{erro(h(\vec{x_n}), y_n)}$$

Se o fator de risco $s_n = \vec{\omega} \cdot \vec{x_n}$ for grande, a chance de um ataque cardíaco é alta. Então, se $y = +1$ (ataque), $e^{-y_n \vec{\omega}\cdot\vec{x_n}}$ será minúsculo, e o erro se aproximará de zero ($\approx ln(1)$)

Já se, com risco alto, o dado resultar em não-ataque ($y = -1$), então o exponencial será positivo, e o erro será enorme

Queremos minimizar esses erros

Regressão Logistica - Minimizando o Erro

Regressão Linear

$$E(\vec{\omega}) = \frac{1}{N}\sum_{n=1}^{N}(\vec{\omega}\cdot\vec{x_n} - y_n)^2$$

Minimizado com o cálculo da pseudo inversa $X^{\dagger} \rightarrow \vec{\omega} = X^{\dagger}\vec{y}$

Regressão Logística

$$E(\vec{\omega}) = \frac{1}{N}\sum_{n=1}^{N} ln(1 + e^{-y_n\vec{\omega}\cdot\vec{x_n}})$$

Como minimizar?

Regressão Logistica - Minimizando o Erro

Embora consigamos derivar uma forma fechada para a solução da regressão linear, o mesmo não é tão facilmente obtido com a regressão logística

Considere seu gradiente (em relação a $\vec{\omega}$)

$$\begin{aligned}
\nabla E(\vec{\omega}) &= \nabla\left(\frac{1}{N}\sum_{n=1}^{N} ln(1+e^{-y_n\vec{\omega}\cdot\vec{x_n}})\right) \\
&= \frac{1}{N}\nabla\left(\sum_{n=1}^{N} ln(1+e^{-y_n\vec{\omega}\cdot\vec{x_n}})\right) \\
&= \frac{1}{N}\sum_{n=1}^{N}\nabla(ln(1+e^{-y_n\vec{\omega}\cdot\vec{x_n}}))
\end{aligned}$$

Regressão Logistica - Minimizando o Erro

$$
\begin{aligned}
\nabla E(\vec{\omega}) &= \frac{1}{N}\sum_{n=1}^{N}\nabla\left(ln(1+e^{-y_n\vec{\omega}\cdot\vec{x_n}})\right) \\
&= \frac{1}{N}\sum_{n=1}^{N}\frac{1}{1+e^{-y_n\vec{\omega}\cdot\vec{x_n}}}\nabla\left(1+e^{-y_n\vec{\omega}\cdot\vec{x_n}}\right) \\
&= \frac{1}{N}\sum_{n=1}^{N}\frac{1}{1+e^{-y_n\vec{\omega}\cdot\vec{x_n}}}e^{-y_n\vec{\omega}\cdot\vec{x_n}}\nabla\left(-y_n\vec{\omega}\cdot\vec{x_n}\right) \\
&= \frac{1}{N}\sum_{n=1}^{N}\frac{1}{1+e^{-y_n\vec{\omega}\cdot\vec{x_n}}}e^{-y_n\vec{\omega}\cdot\vec{x_n}}\left(-y_n\vec{x_n}\right) \\
&= -\frac{1}{N}\sum_{n=1}^{N}\frac{e^{-y_n\vec{\omega}\cdot\vec{x_n}}}{1+e^{-y_n\vec{\omega}\cdot\vec{x_n}}}y_n\vec{x_n}
\end{aligned}
$$

Regressão Logistica - Minimizando o Erro

$$\nabla E(\vec{\omega}) = -\frac{1}{N}\sum_{n=1}^{N}\frac{e^{-y_n\vec{\omega}\cdot\vec{x_n}}}{1+e^{-y_n\vec{\omega}\cdot\vec{x_n}}}y_n\vec{x_n}$$

$$= -\frac{1}{N}\sum_{n=1}^{N}\frac{1}{\frac{1}{e^{-y_n\vec{\omega}\cdot\vec{x_n}}}+1}y_n\vec{x_n}$$

e...

$$\nabla E(\vec{\omega}) = -\frac{1}{N}\sum_{n=1}^{N}\frac{y_n\vec{x_n}}{1+e^{y_n\vec{\omega}\cdot\vec{x_n}}}$$

Regressão Logistica - Minimizando o Erro

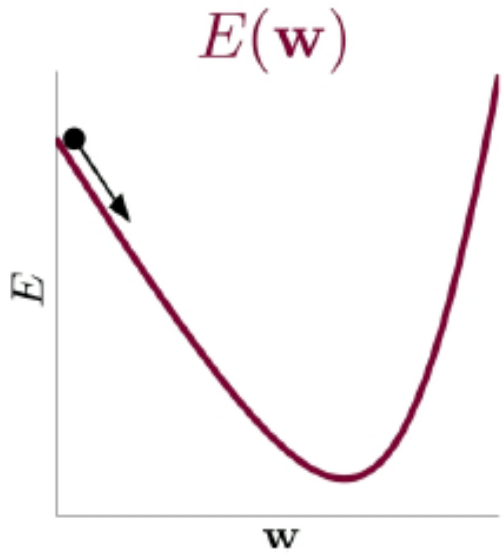
$$\nabla E(\vec{\omega}) = -\frac{1}{N}\sum_{n=1}^{N}\frac{y_n\vec{x_n}}{1+e^{y_n\vec{\omega}\cdot\vec{x_n}}}$$

- Regressão linear
  - forma fechada $\rightarrow$ solução obtida diretamente
- Regressão logística:
  - Essa solução não sai diretamente, por conta da exponencial
  - Solução iterativa $\rightarrow$ vamos melhorando-a passo a passo

Solução Iterativa - Gradient Descent

Vejamos a forma de $E(\vec{\omega})$:

- Forma bastante amigável – há um único mínimo, em vez de vários



Podemos então usar Gradient Descent

- Método geral para otimização não-linear

O método consiste de:

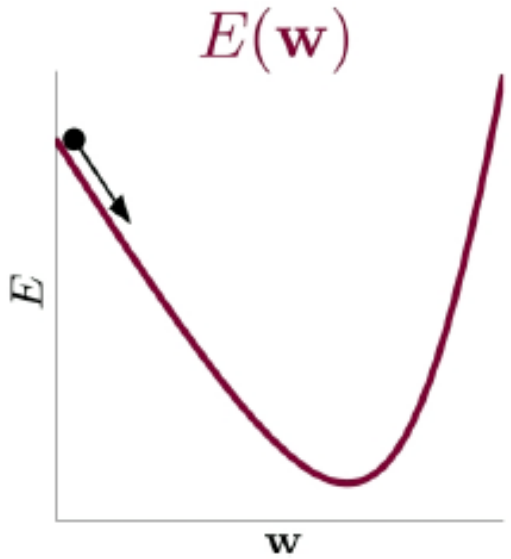
- Iniciar em $\vec{\omega}_0$
- Dar um passo em direção à descida mais íngreme
  - Assumimos que não temos informação global, apenas local $\rightarrow$ não temos como ver o mínimo global da função

Solução Iterativa - Gradient Descent

Fazemos então essa aproximação, nos movendo no espaço de $\vec{\omega}$, em passos de tamanho fixo λ, em uma determinada direção $\hat{v}$ (vetor unitário). Então

$$\vec{\omega}_1 = \vec{\omega}_0 + \lambda \hat{v}$$

Sob essas condições, tentamos descobrir a direção $\hat{v}$



Considere a variação do erro em $\vec{\omega}$:

$$\Delta E(\vec{\omega}) = E(\vec{\omega}_1) - E(\vec{\omega}_0)$$

Gostaríamos que $\Delta E(\vec{\omega})$ fosse o mais negativo possível (queremos minimizar o erro)

Solução Iterativa - Gradient Descent

Considere a expansão de Taylor de uma função $f(x)$ em torno do ponto a:

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \ldots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \ldots$$

Vamos expandir $E(\vec{\omega}_1)$ conforme a série de Taylor em torno do ponto $\vec{\omega}_0$:

$$E(\vec{\omega}_1) = E(\vec{\omega}_0) + \nabla E(\vec{\omega}_0) \cdot (\vec{\omega}_1 - \vec{\omega}_0) + \frac{\nabla^2 E(\vec{\omega}_0)}{2!}|\vec{\omega}_1 - \vec{\omega}_0|^2 + \ldots$$

Solução Iterativa - Gradient Descent

$$E(\vec{\omega}_1) = E(\vec{\omega}_0) + \nabla E(\vec{\omega}_0) \cdot (\vec{\omega}_1 - \vec{\omega}_0) + \frac{\nabla^2 E(\vec{\omega}_0)}{2!} |\vec{\omega}_1 - \vec{\omega}_0|^2 + \ldots$$

Como $\vec{\omega}_1 = \vec{\omega}_0 + \lambda \hat{v} \Rightarrow \vec{\omega}_1 - \vec{\omega}_0 = \lambda \hat{v}$, então

$$E(\vec{\omega}_1) = E(\vec{\omega}_0) + \nabla E(\vec{\omega}_0) \cdot (\lambda \hat{v}) + \frac{\nabla^2 E(\vec{\omega}_0)}{2!} |\lambda \hat{v}|^2 + \ldots$$

Para simplificar, podemos considerar apenas os dois primeiros termos da expansão:

$$E(\vec{\omega}_1) = E(\vec{\omega}_0) + \nabla E(\vec{\omega}_0) \cdot (\lambda \hat{v}) + g(\lambda)$$

(Aproximação de primeira ordem (linear) da superfície)

Solução Iterativa - Gradient Descent

$$E(\vec{\omega}_1) = E(\vec{\omega}_0) + \nabla E(\vec{\omega}_0) \cdot (\lambda \hat{v}) + g(\lambda)$$

A expressão de $\Delta E(\vec{\omega})$ torna-se:

$$\begin{aligned} \Delta E(\vec{\omega}) &= E(\vec{\omega}_1) - E(\vec{\omega}_0) \\ &= E(\vec{\omega}_0) + \nabla E(\vec{\omega}_0) \cdot (\lambda \hat{v}) + g(\lambda) - E(\vec{\omega}_0) \\ &= \nabla E(\vec{\omega}_0) \cdot (\lambda \hat{v}) + g(\lambda) \\ &= \lambda(\nabla E(\vec{\omega}_0)) \cdot \hat{v} + g(\lambda) \end{aligned}$$

A suposição feita pelo Gradient Descent é que podemos ignorar $g(\lambda)$. Então:

$$\Delta E(\vec{\omega}) = \lambda(\nabla E(\vec{\omega}_0)) \cdot \hat{v}$$

Solução Iterativa - Gradient Descent

$$\Delta E(\vec{\omega}) = \lambda(\nabla E(\vec{\omega}_0)) \cdot \hat{v}$$

A questão então é como escolher a direção $\hat{v}$ de modo a fazer $\Delta E(\vec{\omega})$ o mais negativo possível?

A resposta se baseia na observação de que, para qualquer escolha de $\hat{v}$,

$$\lambda(\nabla E(\vec{\omega}_0)) \cdot \hat{v} \geq -\lambda|\nabla E(\vec{\omega}_o)|$$

Como $\hat{v}$ é unitário, então ele não contribui para a magnitude do resultado. Então, o menor valor que podemos conseguir será quando $\hat{v}$ estiver oposto a $\nabla E(\vec{\omega}_0)$, caso em que a magnitude será o tamanho de $\nabla E(\vec{\omega}_0)$, negativo.

Solução Iterativa - Gradient Descent

Assim, se escolhermos $\hat{v}$ em que a igualdade valha, achamos nosso $\hat{v}$, pois é o menor valor que conseguiremos para o erro.

$$\Delta E(\vec{\omega}) = \lambda(\nabla E(\vec{\omega}_0)) \cdot \hat{v} \geq -\lambda|\nabla E(\vec{\omega}_o)|$$

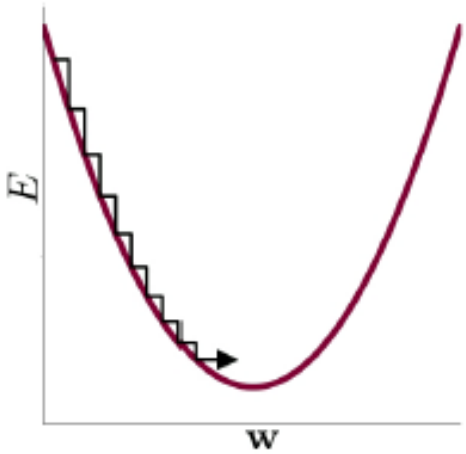
Isso é conseguido escolhendo-se um $\hat{v}$ (unitário) oposto a $\nabla E(\vec{\omega}_o)$:

$$\hat{v} = -\frac{\nabla E(\vec{\omega}_o)}{|\nabla E(\vec{\omega}_o)|}$$

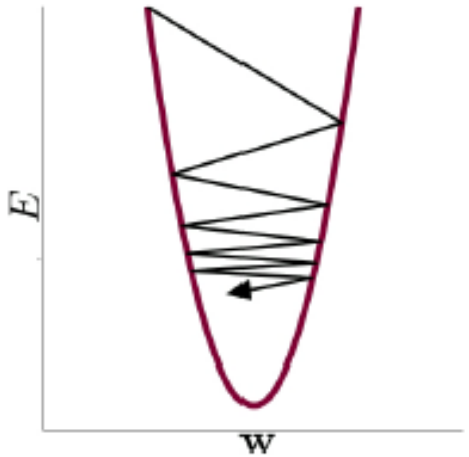
Estamos então descendo ao longo do gradiente do erro. Daí Gradient Descent.

Solução Iterativa - Gradient Descent

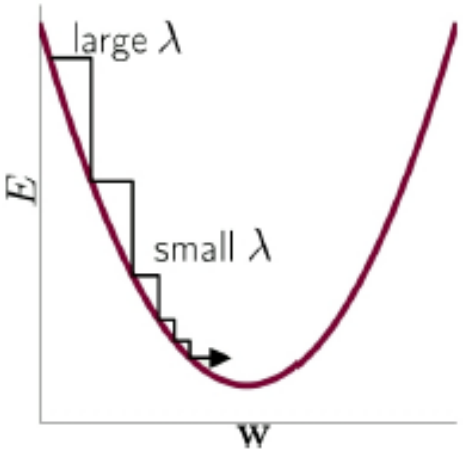
Como, no cálculo do gradiente, mantivemos apenas os termos lineares de Taylor, aproximamos cada passo por uma linha:



λ pequeno demais



λ grande demais



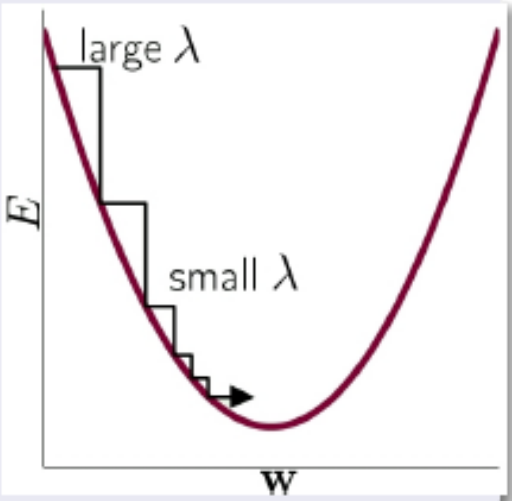
λ de tamanho variável

Embora com passos pequenos estejamos bem próximos à curva, o algoritmo levará muito tempo. Com passos largos, a aproximação poderá ser muito pobre.

Solução Iterativa - Gradient Descent

Podemos ter λ de tamanho variável

- Começamos com um λ grande
- À medida que nos aproximamos ao mínimo o reduzimos
- Assim λ reduziria com a redução da inclinação da curva



Implementação

Comecemos com o passo fixo, lembrando que $\vec{\omega}_1 = \vec{\omega}_0 + \lambda\hat{v}$:

$$\begin{aligned} \Delta\vec{\omega} &= \lambda\hat{v} \\ &= -\lambda\frac{\nabla E(\vec{\omega}_0)}{|\nabla E(\vec{\omega}_0)|} \end{aligned}$$

Solução Iterativa - Gradient Descent

Implementação

Se fizermos λ proporcional a $|\nabla E(\vec{\omega}_0)|$, ou seja, variando conforme o erro, teremos:

$$\lambda = \eta |\nabla E(\vec{\omega}_0)|$$

e

$$\Delta\vec{\omega} = -\lambda \frac{\nabla E(\vec{\omega}_0)}{|\nabla E(\vec{\omega}_0)|} = -\eta |\nabla E(\vec{\omega}_0)| \frac{\nabla E(\vec{\omega}_0)}{|\nabla E(\vec{\omega}_0)|} = -\eta \nabla E(\vec{\omega}_0)$$

Com isso, o passo $\Delta\vec{\omega}$ não é mais fixo, mas sim η, chamada de **taxa de aprendizado**.

Solução Iterativa - Gradient Descent

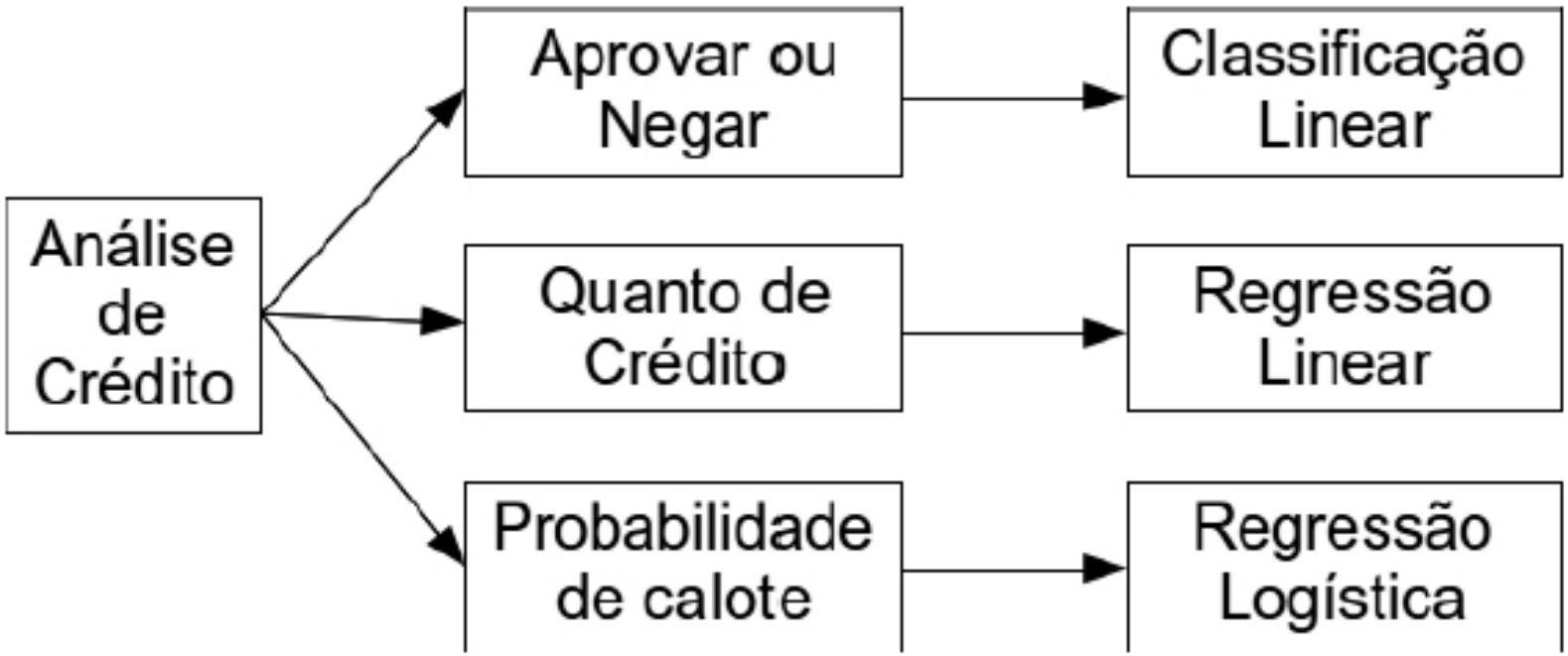
Algoritmo

1. Inicialize os pesos em t_0 com $\vec{\omega}_0$
2. Para $t = 0, 1, 2, \ldots$ faça
3. Calcule o gradiente (slide 68)

$$\nabla E(\vec{\omega}) = -\frac{1}{N}\sum_{n=1}^{N}\frac{y_n\vec{x_n}}{1 + e^{y_n\vec{\omega}_t\cdot\vec{x_n}}}$$

4. Atualize os pesos: $\vec{\omega}_{t+1} = \vec{\omega}_t - \eta\nabla E(\vec{\omega}_t)$
5. Retorne os pesos finais $\vec{\omega}$

Em Suma


Análise de Crédito
Aprovar ou Negar
Quanto de Crédito
Probabilidade de calote
Classificação Linear
Regressão Linear
Regressão Logística

Seleção de Modelos e No Free Lunch Theorem

Em Suma

No contexto do aprendizado de máquina, estamos interessados em modelos que reduzam o erro de regressão ou classificação

- E como testar e comparar? Há vários modos...
  - Validação Cruzada
  - Modelos estatísticos de comparação, como correlação entre os resultados previstos e os reais, por exemplo
  - Etc...
- Serão vistos ao longo do curso

Seleção de Modelos

Qualquer que seja o modelo, alguns passos são necessários

1. Divida aleatoriamente o conjunto de dados em 2 subconjuntos: treinamento e teste
   - Ambos conjuntos precisam ser independentes
   - Nunca treine no de teste
2. Treine o modelo no conjunto de treinamento
3. Teste o modelo no conjunto de teste

Algumas vezes um terceiro conjunto – Validação – é incluído, servindo para ajustes finos do algoritmo durante o treinamento (de modo a evitar *overfitting*

Seleção de Modelos

Uma vez que o teste definitivo se dará em dados reais, é importante que os conjuntos usados para treinamento e teste sejam representativos do todo.

Ainda assim, será que podemos dizer que um método é sempre melhor que outro?

No Free Lunch Theorem

Grupo de teoremas em que se prova, para várias situações, que, de fato, não se consegue aprender gratuitamente apenas olhando as instâncias de treinamento.

- Até porque o mundo real normalmente é bem maior

Seleção de Modelos

Ideia Básica

Se vemos a entrada $(0, 0)$ e $(1, 1)$, ambas classificadas como "falsas", há duas hipóteses (dentre várias) que se adequam aos dados:

- Toda entrada será falsas
- Toda entrada, exceto essas duas, será verdadeira

Ou seja, dado um conjunto de treino, há sempre pelo menos duas generalizações igualmente plausíveis, mas totalmente opostas, que poderiam ser feitas.

- Todo algoritmo precisa de um viés para distinguir entre essas hipóteses
- Ex de viés: aprender apenas alguns tipos de funções (linear etc), introduzir restrições sobre o domínio do problema etc.

No Free Lunch Theorem

Mas fica pior...

Para todo algoritmo α, e para qualquer medida de desempenho, a média sobre todo problema de otimização f (seja função-custo ou função-objetivo) é independente de α

- O que um algoritmo ganha em desempenho em uma classe de problemas é necessariamente contra-balanceado por seu desempenho ruim nos demais problemas

Isso significa que o que se um algoritmo roda bem (em média) em uma classe de funções-objetivo, então ele será pior (em média) nas demais classes.

- Mais que isso, se um algoritmo tem melhor desempenho que seu equivalente aleatório em alguma classe de funções, então terá um desempenho pior nas demais

No Free Lunch Theorem

Ou seja...

Comparações de desempenho de um algoritmo particular, com valores particulares de parâmetros, em uns poucos problemas, são de uso limitado.

- Não existe modelo que funcione melhor para todo problema.
- O pressuposto de um ótimo modelo para um problema pode não valer para outro.
- É comum, em aprendizado, tentar vários modelos e ver qual o que funciona melhor para um problema particular.

No Free Lunch Theorem

e...

Da mesma forma que não há algoritmo universalmente eficaz, não há problema f, tanto de busca quanto otimização, que possa resultar em desempenho melhor que o aleatório, independentemente do algoritmo usado (ou seja, para todo e qualquer algoritmo usado).

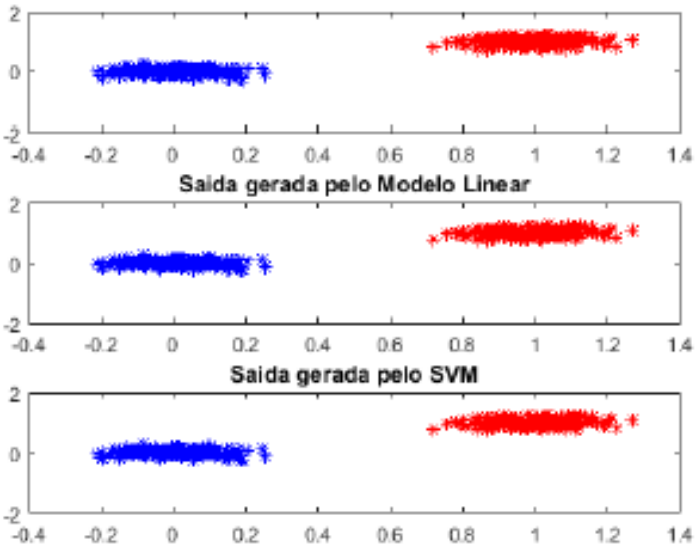
Resumindo

Não existem modelos genéricos de aprendizado de máquina que, em média, apresentem melhor desempenho que qualquer outro modelo para uma classe de problemas quaisquer.

Comparação entre Modelo Linear versus SVM com Kernel Linear

Experimento # 1

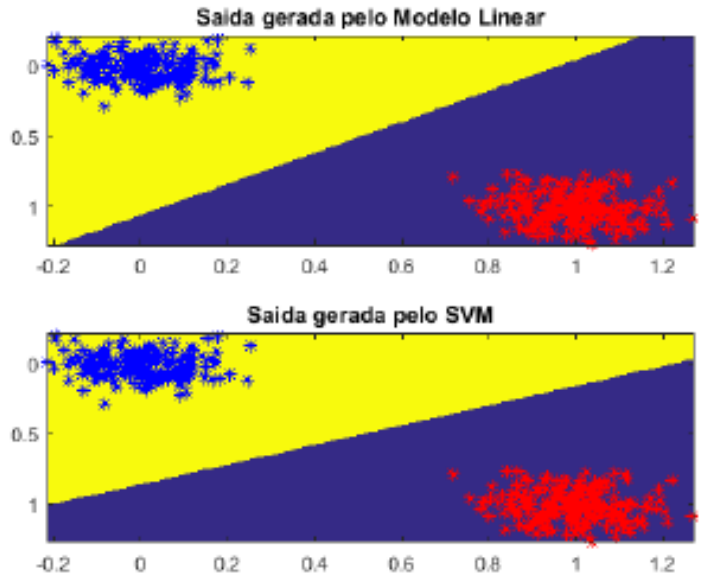
Considere duas distribuições uniformes com centro [0,0] e [1,1], ambas com desvio padrão [0.1, 0.1].



Comparação entre Modelo Linear versus SVM com Kernel Linear

Experimento # 1

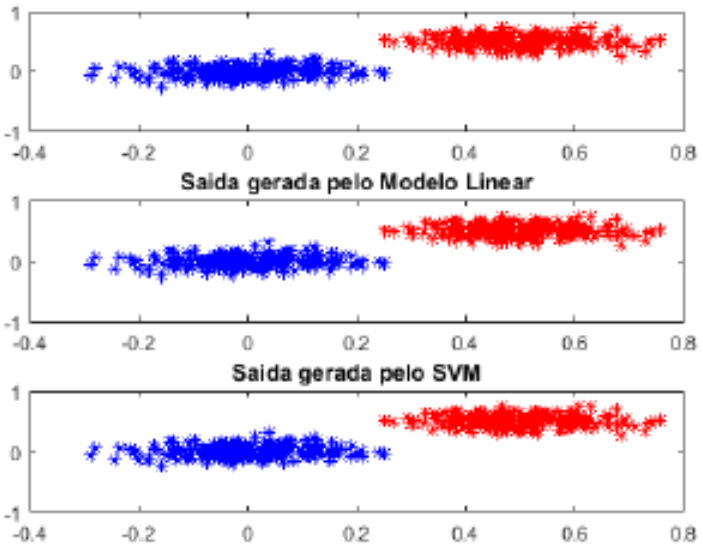
Fronteira de decisão



Comparação entre Modelo Linear versus SVM com Kernel Linear

Experimento # 2

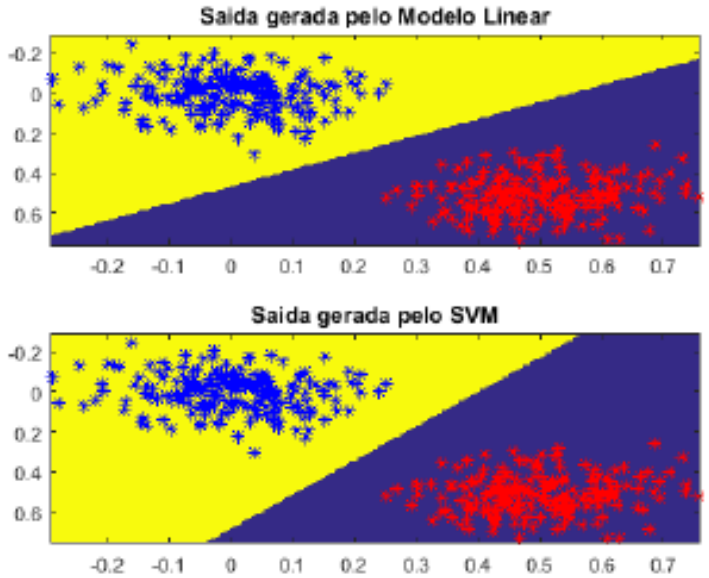
Considere duas distribuições uniformes com centro [0,0] e [0.5,0.5], ambas com desvio padrão [0.1, 0.1].



Comparação entre Modelo Linear versus SVM com Kernel Linear

Experimento # 2

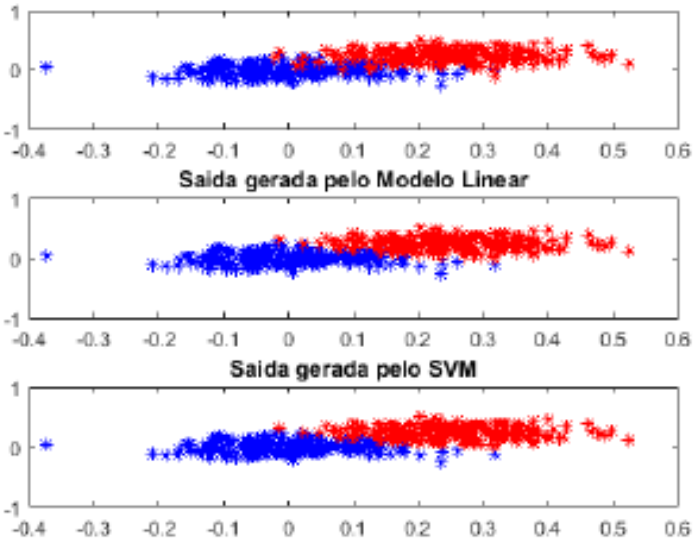
Fronteira de decisão



Comparação entre Modelo Linear versus SVM com Kernel Linear

Experimento # 3

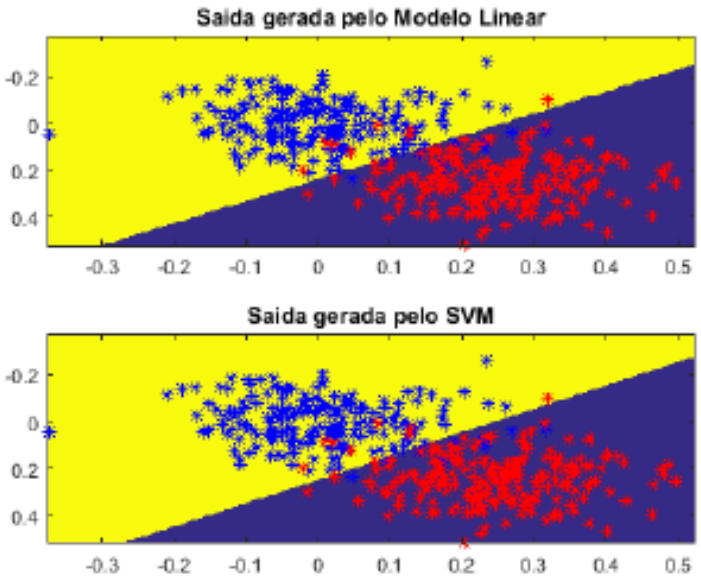
Considere duas distribuições uniformes com centro [0.25,0.25], ambas com desvio padrão [0.1, 0.1].



Comparação entre Modelo Linear versus SVM com Kernel Linear

Experimento # 3

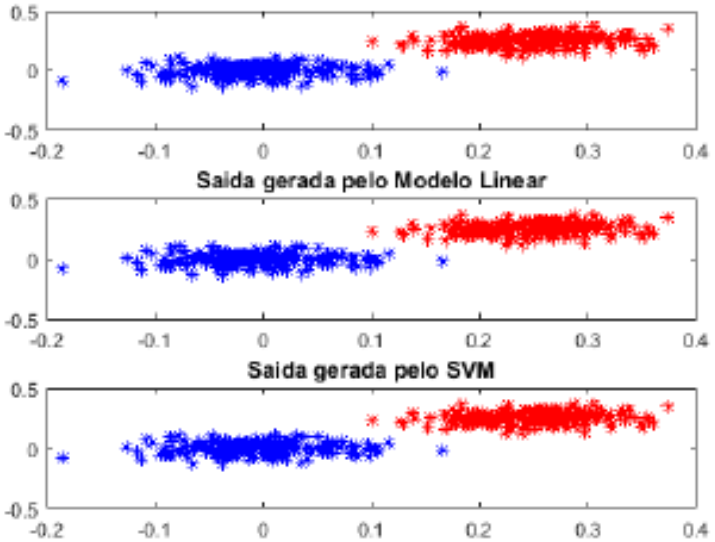
Fronteira de decisão



Comparação entre Modelo Linear versus SVM com Kernel Linear

Experimento # 4

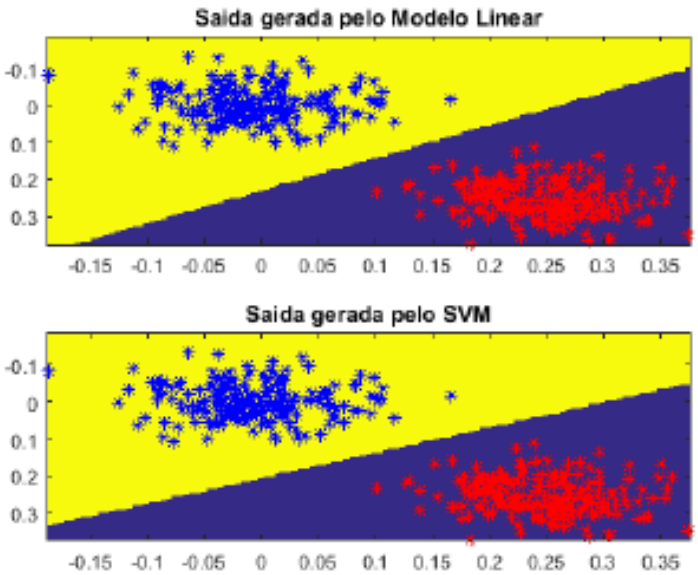
Considere duas distribuições uniformes com centro [0,0] e [0.25,0.25], ambas com desvio padrão [0.05, 0.05]. .



Comparação entre Modelo Linear versus SVM com Kernel Linear

Experimento # 4

Fronteira de decisão



Referências

- Alpaydın, E.: Introduction to Machine Learning. 2 ed. MIT Press. 2010.
- Harrington, P.: Machine Learning in Action. Manning. 2012.
- Ho, Y.C.; Pepyne, D.L.: Simple Explanation of the No Free Lunch Theorem of Optimization. Em Proceedings of the 40th IEEE Conference on Decision and Control. Orlando, FL, EUA. Dezembro, 2001.
- Machine Learning, Neural and Statistical Classification. D. Michie, D.J. Spiegelhalter, C.C. Taylor (eds.). 1994.
- Murphy, K. P.: Machine Learning: A Probabilistic Perspective. MIT Press. 2012.
- Wolpert, D. H.: The lack of a priori distinctions between learning algorithms. Em Neural Computation 8(7). Outubro, 1996.
- Wolpert, D.H.: What the No Free Lunch Theorems Really Mean: How to Improve Search Algorithms. SFI WORKING PAPER: 2012-10-017. Santa Fe Institute. Maio, 2012.

Referências

- Wolpert, D. H.; Macready, W. G.: No Free Lunch Theorems for Optimization. Em IEEE Transactions On Evolutionary Computation 1(1). Abril, 1997.
- http://work.caltech.edu/library/
- http://www.cbcb.umd.edu/~hcorrada/PracticalML/pdf/lectures/
- http://www.aihorizon.com/essays/generalai/no_free_lunch_machine_learning.htm
- http://www.statsblogs.com/2014/01/25/machine-learning-lesson-of-the-day-the-no-free-lunch-theorem/