

Aprendizado de Máquina

# **Regressão**

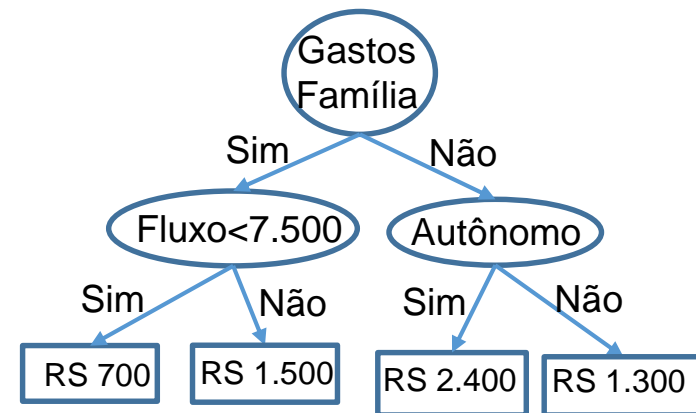
Eduardo R. Hruschka

# Agenda

- Árvores
- Random Forests
- k-Nearest Neighbors
- LASSO (Least Absolute Shrinkage and Selection Operator)
- Noções sobre Redes Neurais
- Avaliação de Modelos

# Recapitulando – modelagem via árvores

- Procedimento idêntico àquele para construir árvores de classificação, exceto pela função objetivo de otimização:
  - Trocar entropia pelo erro quadrático;
- Particionamento recursivo dos dados;
- Considerar todos os valores de todas as variáveis na construção do modelo;
- Selecionar par de variável-valor que produza o melhor ajuste à variável dependente;
- Repetir o processo em cada um dos nós, gerando uma árvore.

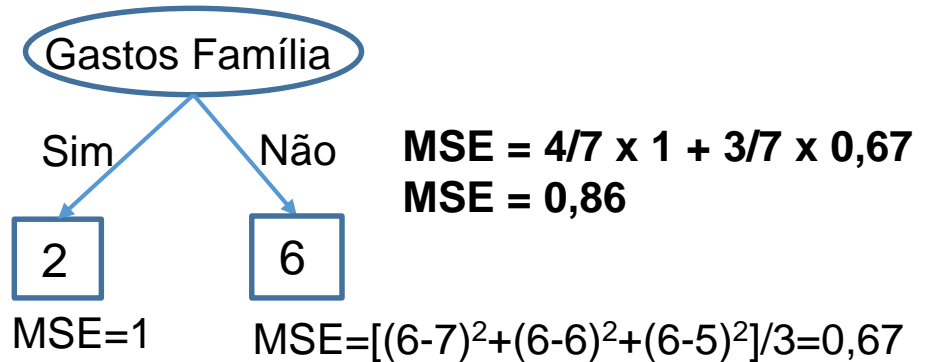


# Árvores de Regressão - exemplo

Gastos Família	Autônomo	Receita (R\$ x 10 <sup>3</sup> )
Sim	Sim	1
Sim	Não	3
Sim	Sim	1
Sim	Não	3
Não	Sim	7
Não	Não	6
Não	Sim	5

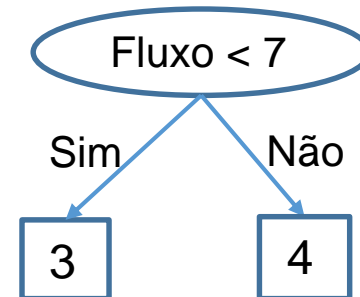
$$\mu = 3,7$$

$$\text{MSE} = [(3,7-1)^2 + (3,7-3)^2 + \dots + (3,7-5)^2] / 7$$
$$\text{MSE} = 4,78$$



## Exercício:

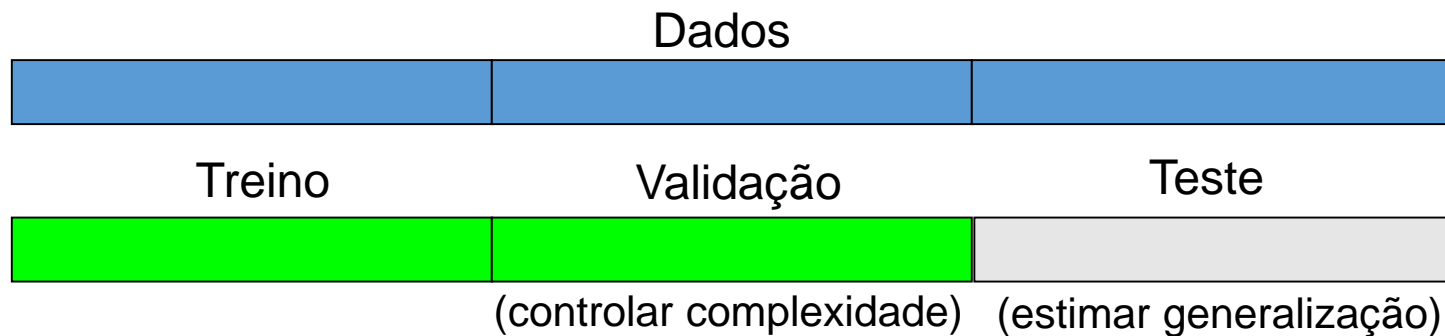
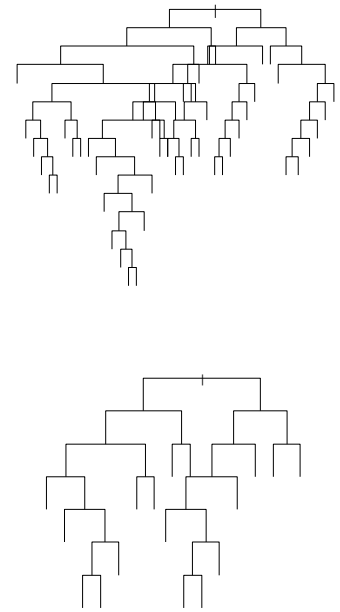
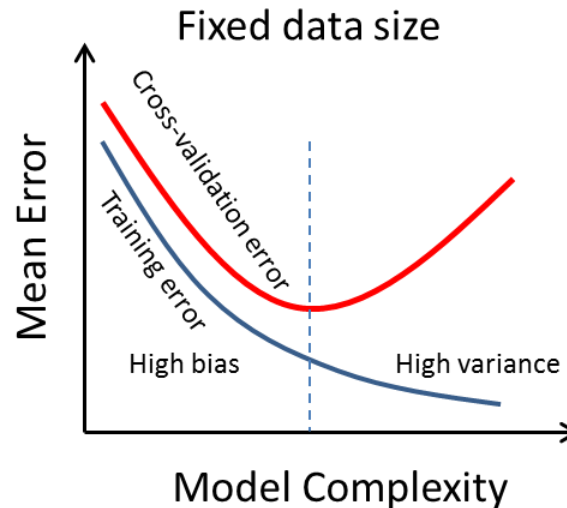
- Realizar o teste para o atributo "Autônomo".



- Após testar todos os atributos, escolhe-se o melhor (MSE);
- Atributos contínuos e ordinais são ordenados e testados por pontos de corte:

# Cuidado com o superajuste

- Uma abordagem prática envolve controlar via validação cruzada, separando parte do conjunto de treinamento para otimizar a profundidade da árvore:



# Vantagens das árvores de regressão

- Modelo não paramétrico (não presume f.d.p!);
- Realiza seleção de atributos automaticamente;
- Lida com atributos binários, categóricos, ordinais e contínuos;
- Descobrir regras que mostram interações entre variáveis;
- Lida com valores ausentes;
- Pouco sensível a outliers na variável dependente;
- Útil para análise exploratória de dados;
- Serve para categorizar variáveis – e.g., idade (jovem, adulto, velho) em função da correlação com a variável dependente (e.g., renda).

# Desvantagens das árvores de regressão

- Modelo obtido é uma “*step function*”;
    - Pode ser melhorado com regressores lineares nos nós folha.
  - Problemas complexos requerem árvores complexas;
    - Perde-se a interpretabilidade.
  - Modelos instáveis
    - Flutuações amostrais podem resultar em árvores bem diferentes;
- Problemas difíceis? Plante uma floresta!
- *Random Forests* ou *Boosting*.

Quiz:



# Agenda

- Árvores
- Random Forests
- k-Nearest Neighbors
- LASSO (Least Absolute Shrinkage and Selection Operator)
- Noções sobre Redes Neurais
- Avaliação de Modelos



- Framework idêntico àquele visto em classificação:

### **Indução do modelo de regressão:**

Considere que temos  $N$  exemplos de treinamento.

Para cada uma das  $t$  iterações faça:

- 1 - Amostrar  $N$  exemplos com reposição (*bagging*).
- 2 - Induzir uma árvore usando apenas um subconjunto aleatoriamente escolhido dos atributos (e.g.,  $m \leq n^{1/2}$ ).
- 3 - Armazenar a árvore obtida.

### **Predição**

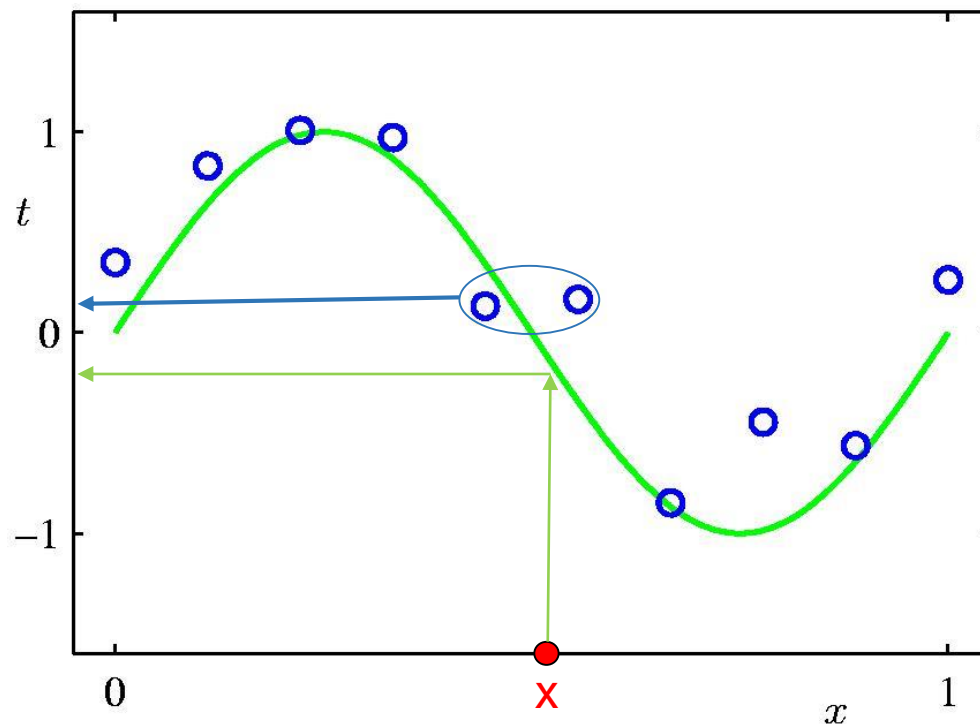
Para cada uma das  $t$  árvores:

Fazer a predição para o exemplo do conjunto-alvo.  
Retornar a média das predições realizadas.

# Agenda

- Árvores
- Random Forests
- k-Nearest Neighbors
- LASSO (Least Absolute Shrinkage and Selection Operator)
- Avaliação de Modelos

# Noção Intuitiva



Dado  $x$ ,  $t = f(x)$  ?  
Problema:  $f(x)$  é desconhecida.

Solução k-NN: Estimar  $t$  por meio da média dos vizinhos mais próximos (em relação aos valores de  $x$  apenas).

$$Erro = f(x) - \hat{t}$$

- K-NN estende essa ideia simples para espaços multidimensionais;
- Abordagem praticamente idêntica àquela vista em classificação.

# Conceitos fundamentais

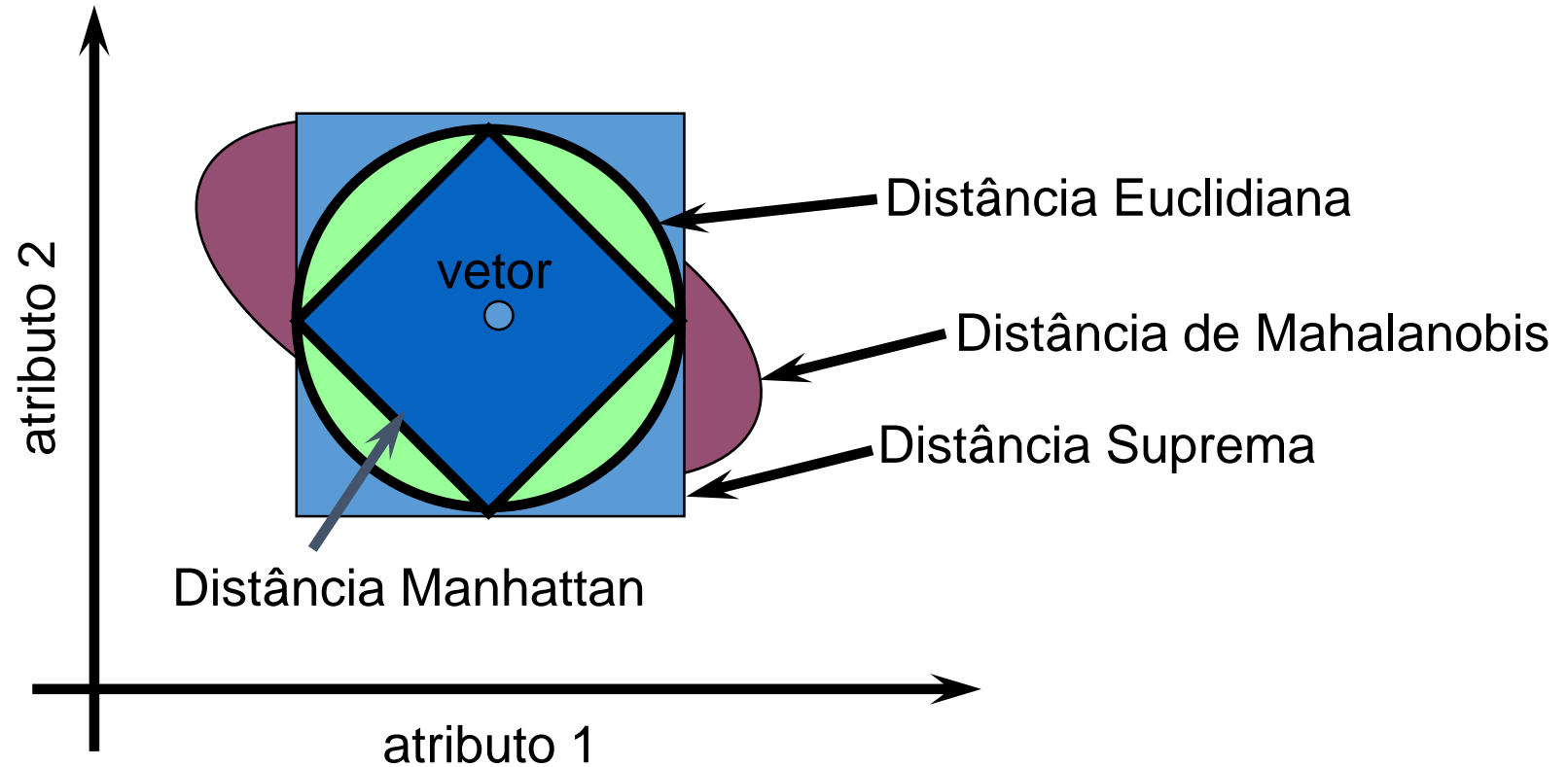
- Exemplos correspondem a pontos no  $\mathbb{R}^n$ ;
- Vizinhos definidos em função de uma medida de distância;
- Por exemplo, considerando-se dois vetores  $\mathbf{x}=[x_1, x_2, \dots, x_n]$  e  $\mathbf{y}=[y_1, y_2, \dots, y_n]$ , a distância Euclidiana entre estes é:

$$d_E(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

- Há várias outras medidas de distância, e.g.:
  - *Simple matching approach* (vista em classificação);
  - Outra bastante popular é a Mahalanobis (tarefa de casa opcional), particularmente útil quando os dados foram previamente agrupados:
    - Agrupar dados (variáveis independentes) via k-means;
    - Computar distâncias para os centroides dos *clusters* apenas.

# Grupos induzidos por diferentes distâncias

Onde se situam os pontos equidistantes de um vetor?



# K-NN para regressão

## Algoritmo:

Dado um exemplo  $\mathbf{x}_q$  cujo valor da variável dependente ( $y$ ) se deseja estimar e considerando que  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$  representam os  $k$  exemplos mais próximos de  $\mathbf{x}_q$ , retornar:

$$y = f(\mathbf{x}_q) = \frac{\sum_{i=1}^k f(\mathbf{x}_i)}{k}$$

- Predição por meio da média da vizinhança
- **K-NN ponderado pelo inverso da distância:**

$$y = f(\mathbf{x}_q) = \frac{\sum_{i=1}^k w_i f(\mathbf{x}_i)}{\sum_{i=1}^k w_i}$$

$$w_i = \frac{1}{d(\mathbf{x}_q, \mathbf{x}_i)^2}$$

# Parâmetros?

- Normalização / padronização ?
- Medida de distância ?
- Número de vizinhos,  $k = ?$
- Pesos dos atributos ?
- Preprocessamento via clustering minimiza custo de inferência, bem como custo de otimização de parâmetros.

Validação  
Cruzada

➤ k-NN é muito usado na prática por conta da simplicidade de implementação computacional, mas custo de inferência pode ser fator impeditivo.

# Exercício

Considere a seguinte base de dados:

<b>Exemplo</b>	<b>a<sub>1</sub></b>	<b>a<sub>2</sub></b>	<b>a<sub>3</sub></b>	<b>Y</b>
<b>1</b>	0	250	36	10
<b>2</b>	10	150	34	15
<b>3</b>	2	90	10	5
<b>4</b>	6	78	8	20
<b>5</b>	4	20	1	30
<b>6</b>	1	170	70	40
<b>7</b>	8	160	41	25
<b>8</b>	10	180	38	35
<b>9</b>	6	200	45	?

- Estimar Y para o objeto #9 com  $k=1,2,3,4,5$  (com e sem ponderação)

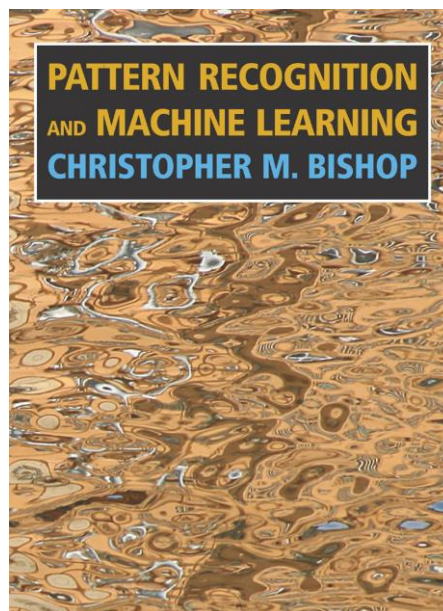


# Agenda

- Árvores
- Random Forests
- k-Nearest Neighbors
- LASSO (Least Absolute Shrinkage and Selection Operator)
- Noções sobre Redes Neurais
- Avaliação de Modelos

## Para atingir nosso objetivo (LASSO):

- Presumir conhecimento de regressão linear;
- Prover uma visão geral sobre modelos lineares;
- Perspectiva baseada em funções de base;
- Leitura recomendada:



# Modelos lineares de funções de base

- Consideremos o modelo de regressão linear:

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_D x_D$$

- Um modelo mais geral é:

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$$

- Note que  $M$  é o número de parâmetros do modelo e que para a regressão linear temos  $M = D + 1$ :

$$\phi_0(\mathbf{x}) = 1, \phi_1(\mathbf{x}) = x_1, \dots, \phi_{M-1}(\mathbf{x}) = x_D$$

- $\boldsymbol{\phi}(\mathbf{x})$  são as funções de base (Gaussiana, polinômio, sigmoid etc.) para um vetor  $\mathbf{x}$ .

**Recapitulação.** Presumindo observações de uma função determinística com ruído Gaussiano:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \quad \text{onde} \quad p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$$

- Equivalente a dizer que ( $\beta=1/\sigma$ , i.e. a *precisão*):

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}).$$

- Dados *inputs*  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  e valores-alvo  $\mathbf{t} = [t_1, \dots, t_N]^T$  obtemos a função de verossimilhança:

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^N \mathcal{N}(t_n|\mathbf{w}^T \phi(\mathbf{x}_n), \beta^{-1}).$$

Tomando seu logaritmo temos:

$$\begin{aligned}\ln p(\mathbf{t}|\mathbf{w}, \beta) &= \sum_{n=1}^N \ln \mathcal{N}(t_n | \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \\ &= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathbf{w})\end{aligned}$$

Onde

$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

é a soma dos erros quadráticos, sendo este o único termo que depende de  $\mathbf{w}$ ...

Computando o gradiente e igualando-o a **zero** temos:

$$\nabla_{\mathbf{w}} \ln p(\mathbf{t}|\mathbf{w}, \beta) = \beta \sum_{n=1}^N \{t_n - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n)\} \boldsymbol{\phi}(\mathbf{x}_n)^T = \mathbf{0}.$$

Resolvendo para  $\mathbf{w}$  obtemos:

$$\mathbf{w}_{\text{ML}} = \left( \boldsymbol{\Phi}^T \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^T \mathbf{t}$$

Onde:

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}.$$

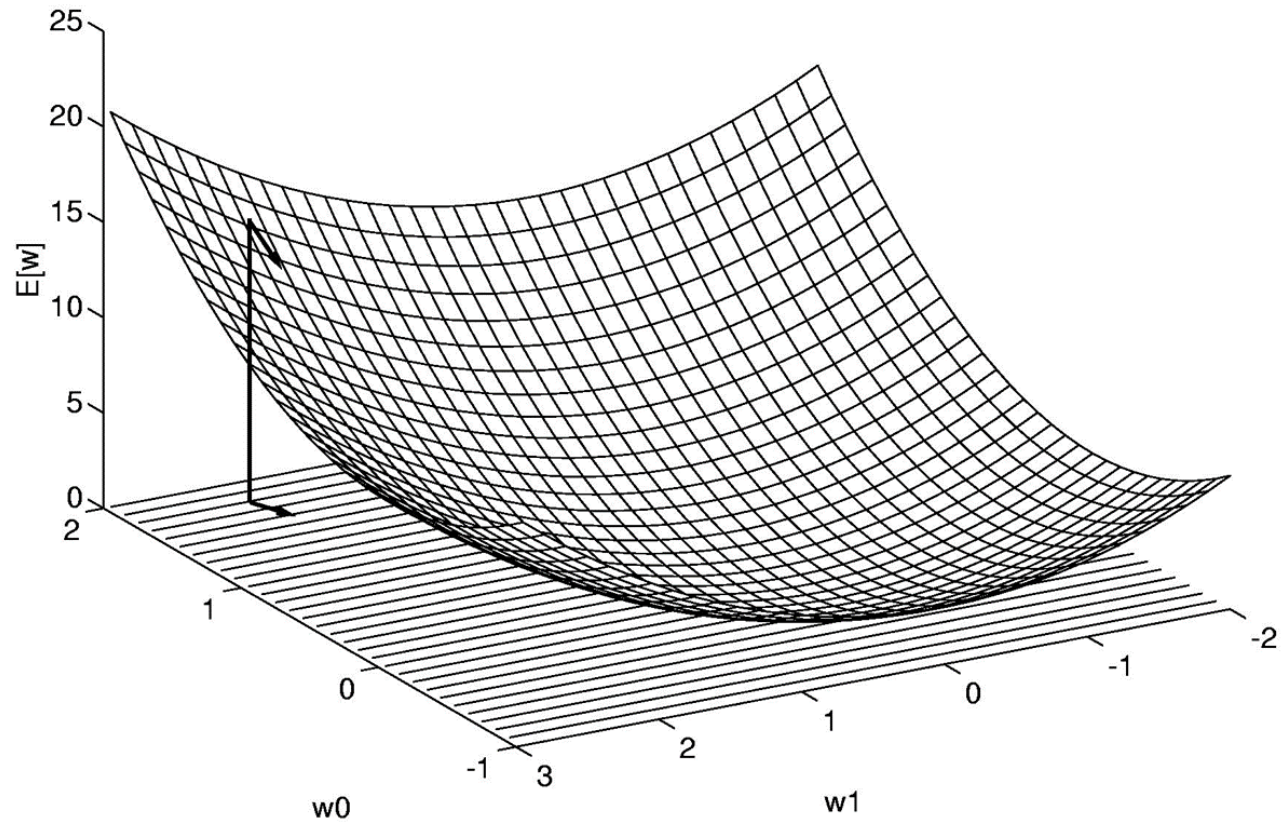
# Aprendizado sequencial de parâmetros

- Considere agora que os itens de dados sejam disponibilizados individualmente ao longo do tempo (cenário de aprendizado online / fluxo de dados);
- Podemos usar a técnica (geral) de *gradiente descendente estocástico* para estimar parâmetros:

$$\begin{aligned}\mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} - \eta \nabla E_n \\ &= \mathbf{w}^{(\tau)} + \eta (t_n - \mathbf{w}^{(\tau)\top} \phi(\mathbf{x}_n)) \phi(\mathbf{x}_n).\end{aligned}$$

- Iniciar com pesos aleatórios;
- Como escolher a taxa de aprendizado ( $\eta$ ) ?
  - Precisa ser suficientemente baixo, e.g.,  $\eta = 0,1$ .
  - Observar comportamento de  $E_D(\mathbf{w})$ .

# Gradient Descent





## Regularização (a.k.a *shrinkage*)

- Sabe-se que o *overfitting* tende a aumentar  $\|\mathbf{w}\|$ .
- Para obter melhor capacidade de generalização do modelo consideramos uma função de erro na forma:

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

Com um regularizador quadrático queremos minimizar:

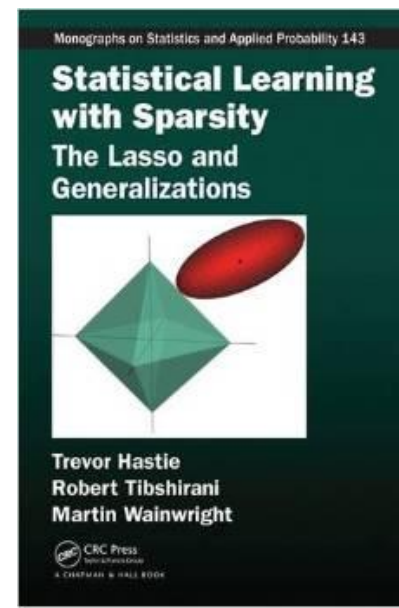
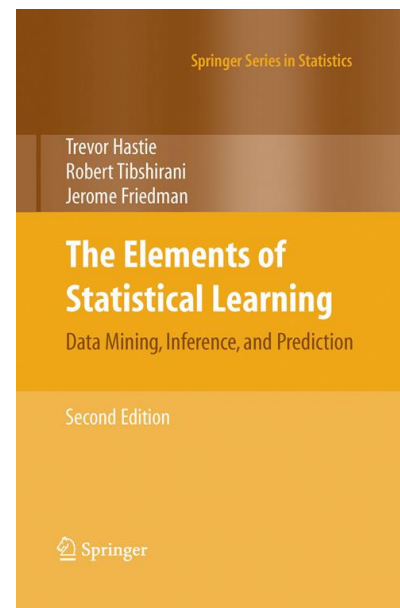
$$\frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2} \mathbf{w}^T \mathbf{w}$$

Onde  $\lambda$  é o coeficiente de regularização. Os parâmetros do modelo passam a ser estimados via (*Ridge Regression*):

$$\mathbf{w} = \left( \lambda \mathbf{I} + \Phi^T \Phi \right)^{-1} \Phi^T \mathbf{t}.$$

# Regularização via LASSO

- *Ridge Regression* funciona bem quando há um subconjunto dos pesos (verdadeiros) pequenos/zerados;
- Entretanto, coeficientes não são zerados; logo não é completamente apropriada para realizar seleção de atributos;
- Essa desvantagem em geral não prejudica a capacidade preditiva, mas pode complicar a interpretabilidade;
- Adotar outra regularização (norma para punir os pesos);
- LASSO (Least Absolute Shrinkage and Selection Operator) foi proposto por Tibshirani em 1994.



# Regularização via LASSO

- Utilizando uma expressão mais geral, ao adotarmos  $q=1$  obtemos a função objetivo do LASSO:

$$\frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2} \sum_{j=1}^M |w_j|^q$$

- Note que com  $\lambda=0$  obtemos os pesos de uma regressão linear, enquanto que com  $\lambda=\infty$  temos  $\mathbf{w}=\mathbf{0}$ .
- Penalidade faz com que alguns pesos sejam iguais a zero: induz modelos esparsos e (mais) interpretáveis.
- Padronizar variáveis de tal forma que tenham variância unitária, e.g., via escore z.

# Algoritmo de aprendizado

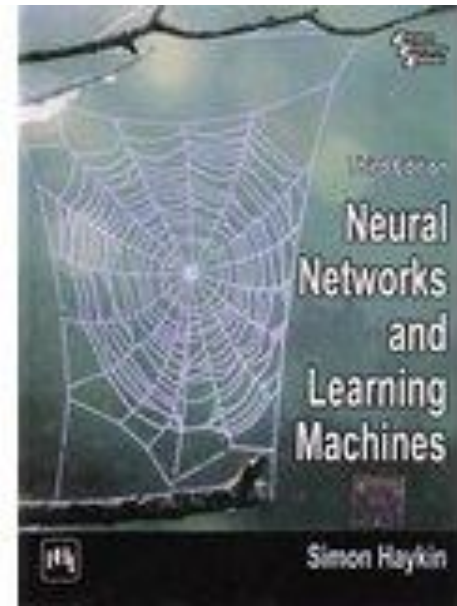
- Implementação usa programação quadrática
- Complexidade de tempo aproximadamente cúbica com o número de variáveis

# Agenda

- Árvores
- Random Forests
- k-Nearest Neighbors
- LASSO (Least Absolute Shrinkage and Selection Operator)
- Noções sobre Redes Neurais
- Avaliação de Modelos

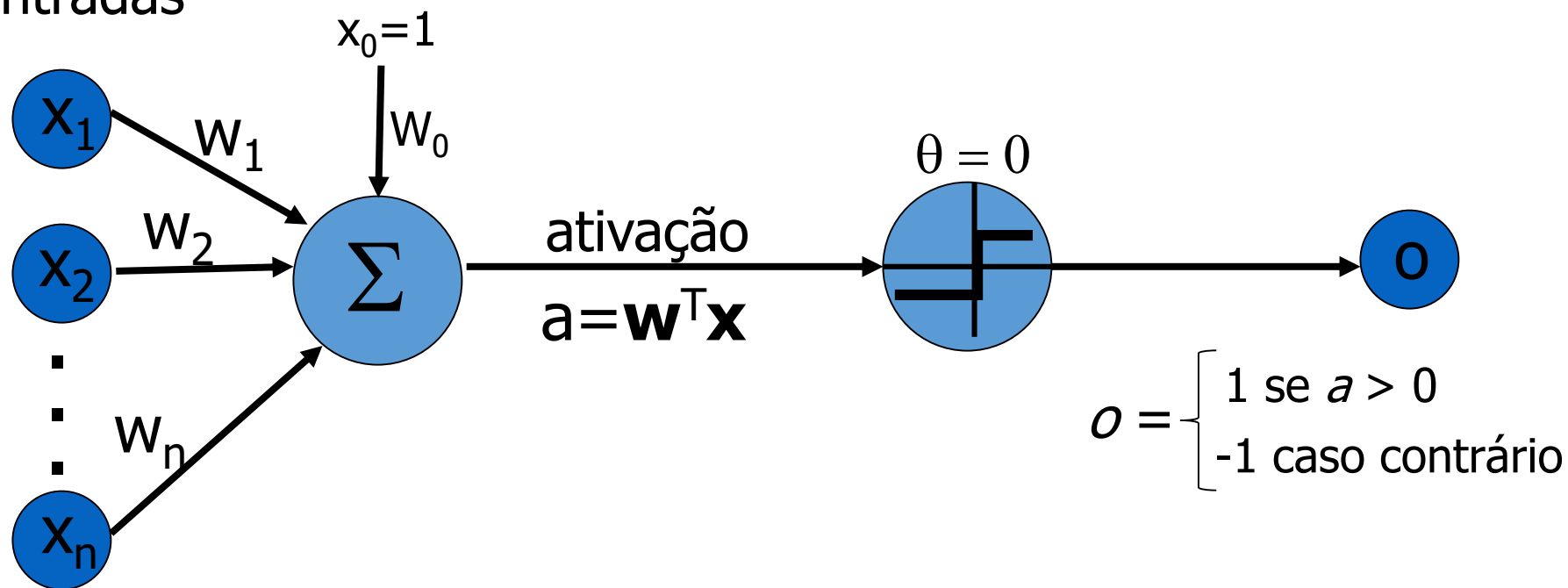
# Noções sobre Redes Neurais - Motivação

- Funcionamento do cérebro humano, que é o sistema de aprendizado mais robusto que conhecemos;
- Explorar paralelismo massivo;
- *Perceptron*: algoritmo para aprender redes neurais simples, de uma única camada (década de 50);
- *Backpropagation*: algoritmo para treinar redes de múltiplas camadas que foi desenvolvido nos anos 80;
- Base conceitual para *deep learning*;
- Sugestão de leitura: livros de Simon Haykin.



# Perceptron (Rosenblatt, 1957)

entradas

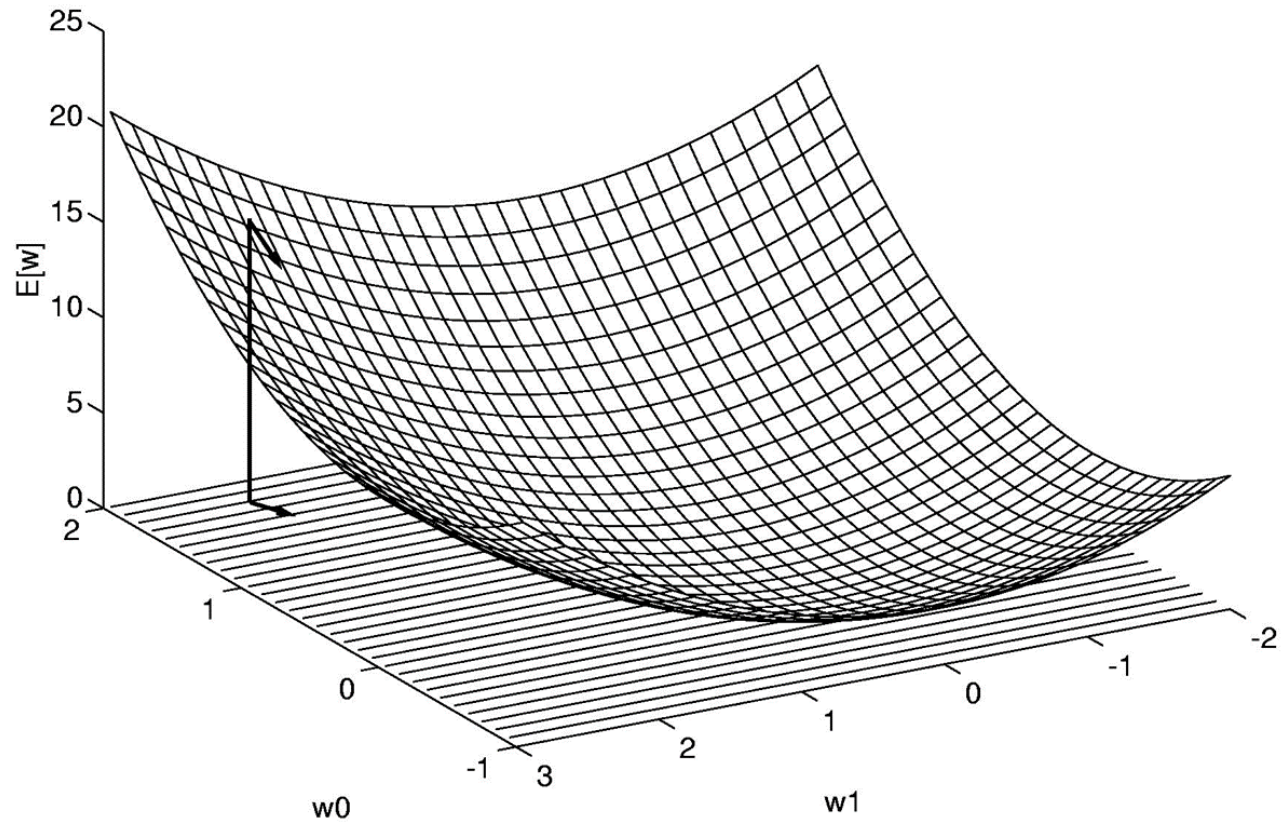


Considere que:

- $t$  é o valor alvo
- $\eta$  é a taxa de aprendizado
- $o = a = \mathbf{w}^T \mathbf{x}$  (por simplicidade)

➤ Minimizar:  $E(\mathbf{w}) = \frac{1}{2} \sum (t - o)^2$

# Gradient Descent





Gradient:

$$\nabla E[\vec{w}] \equiv \left[ \frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \dots, \frac{\partial E}{\partial w_n} \right]$$

Training rule:

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}]$$

I.e.:

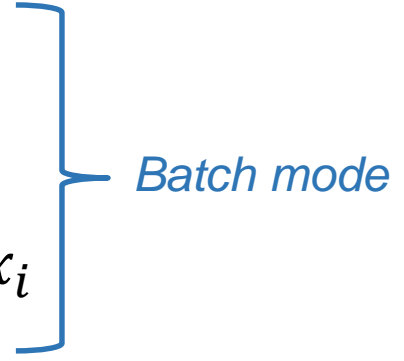
$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

# Gradient Descent

$$\begin{aligned}\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i} \frac{1}{2} \sum_d (t_d - o_d)^2 \\&= \frac{1}{2} \sum_d \frac{\partial}{\partial w_i} (t_d - o_d)^2 \\&= \frac{1}{2} \sum_d 2(t_d - o_d) \frac{\partial}{\partial w_i} (t_d - o_d) \\&= \sum_d (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - \vec{w} \cdot \vec{x}_d) \\ \frac{\partial E}{\partial w_i} &= \sum_d (t_d - o_d)(-x_{i,d})\end{aligned}$$

## Algoritmo:

Entradas: exemplos de treinamento ( $d \in D$ ) e  $\eta$ .

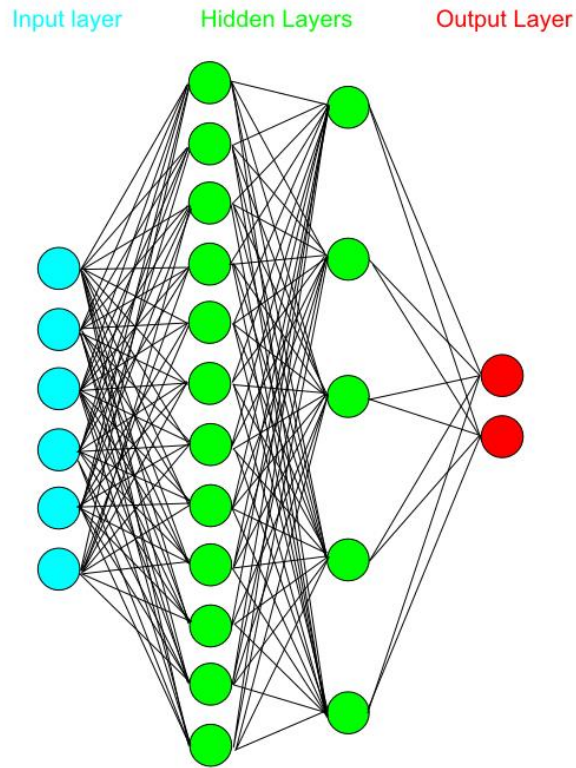
1. Inicializar aleatoriamente cada peso  $w_i$  ( $\mathbf{w}$  versor)
  2. Até que uma condição de término seja atingida fazer:
    - a. Inicializar  $\Delta w_i = 0$
    - b. Para cada  $(\mathbf{x}, t)$  em  $D$  fazer:
      - Normalizar  $\mathbf{x}$  e computar  $o$
      - Computar  $\Delta w_i \leftarrow \Delta w_i + \eta(t - o)x_i$
    - c. Para cada peso fazer:
      - $w_i \leftarrow w_i + \Delta w_i$
- 
- Batch mode*

# Gradiente descendente

- Converge para os pesos de menor erro quadrático para  $\eta$  suficientemente pequeno, mesmo com dados ruidosos;
- Exemplos de treinamento precisam ser linearmente separáveis;
- Gradiente descendente incremental (estocástico) converge para gradiente em lote para  $\eta$  suficientemente pequeno;

**Nota:** a diferença entre o gradiente em lote e o estocástico se refere a agregar (ou não) as atualizações dos pesos.

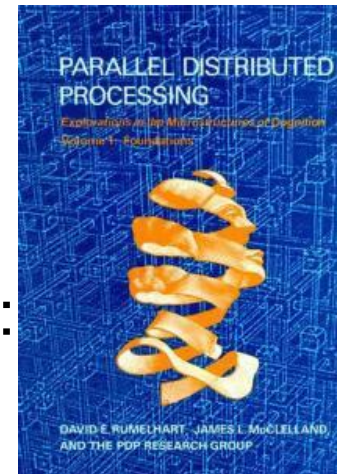
# Multilayer perceptrons (MLP)



- Surgiram para superar limitações do perceptron simples;
- Minsky & Papert (1969) demonstraram limitações do perceptron; especularam que limitações não poderiam ser superadas;

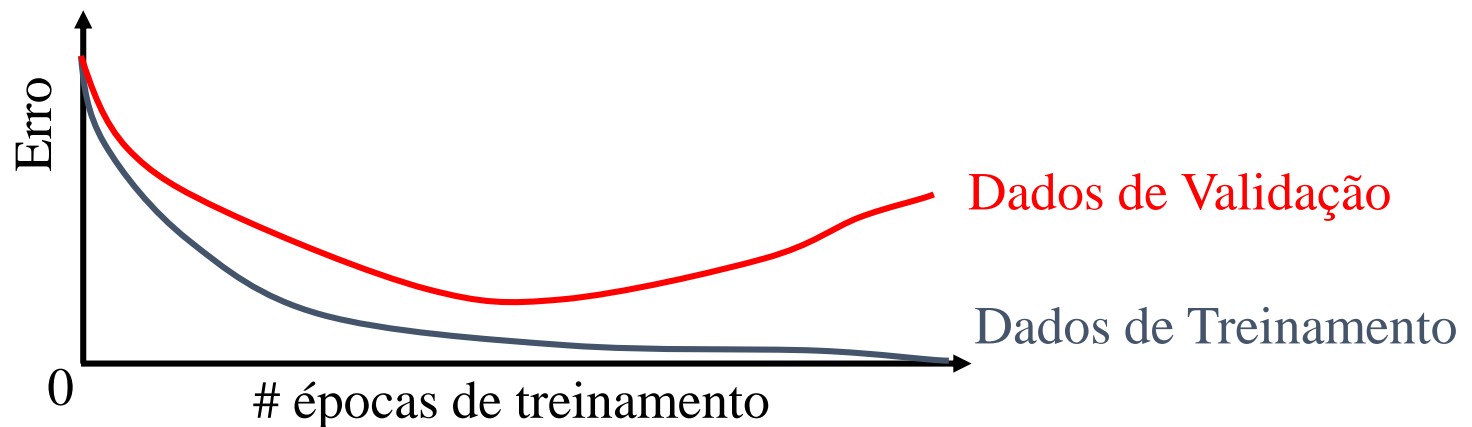


- LeCun (1985) e colegas mostraram que estavam errados: backpropagation.
- Ver também Rumelhart e McClelland:



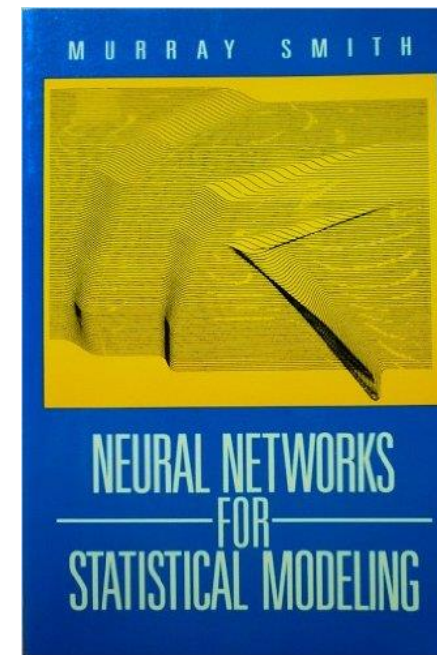
# Algoritmo de *backpropagation*

- Ideia de treinamento baseada no perceptron;
- Obter ótimo local (normalmente suficiente na prática);
- Treinar várias redes, a partir de arquiteturas e inicializações diferentes;
- Computacionalmente intenso no treinamento; rápida para fazer inferências;
- Cuidar com *overfitting*:



# Capacidade de aprendizado de MLPs

- Toda função contínua pode ser aproximada, com um erro arbitrariamente pequeno, por um MLP de uma camada oculta;
- MLP com duas camadas ocultas aproxima, com acurácia arbitrária, qualquer função;
- Por que então deveríamos nos preocupar com outros algoritmos?
  - Superajuste, *bias* x variância, tempo de treinamento, interpretabilidade.
  - Literatura muito focada na teoria. Para uma visão prática, ver livro de Murray Smith (1996): transformações, funções de ativação, inicialização dos pesos etc.



# Agenda

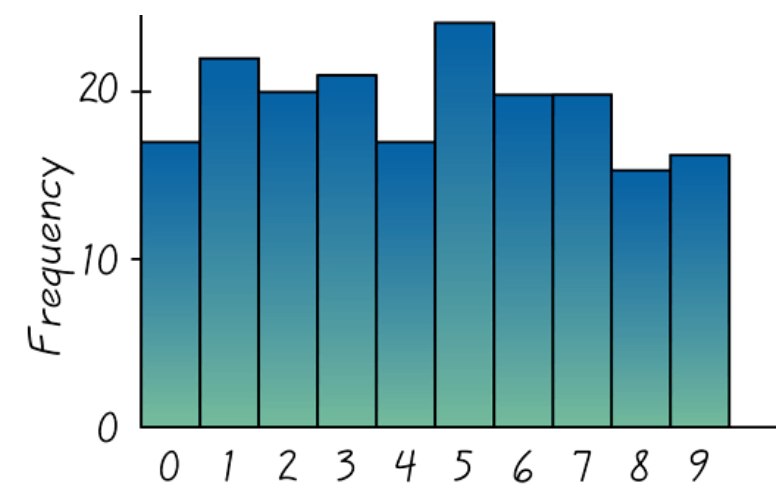
- Árvores
- Random Forests
- k-Nearest Neighbors
- LASSO (Least Absolute Shrinkage and Selection Operator)
- Noções sobre Redes Neurais
- Avaliação de Modelos



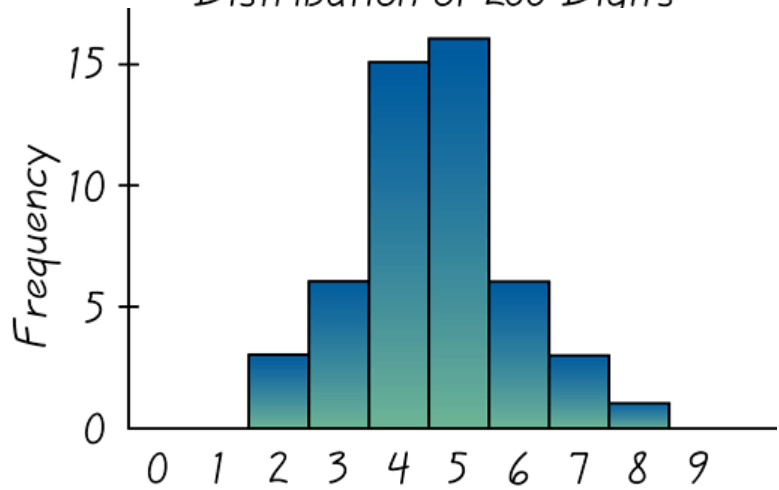
# Avaliação de modelos (material para estudo extra classe)

1. Breve revisão sobre testes de significância;
2. Comparando dois algoritmos;
3. Comparando vários algoritmos (leitura).

# Aquecimento – Teorema Central do Limite



Distribution of 200 Digits



Distribution of 50 Sample Means

$$\mu_{\bar{x}} = \mu$$
$$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}}$$

Table 5-6					
SSN digits				$\bar{x}$	
1	8	6	4	4.75	
5	3	3	6	4.25	
9	8	8	8	8.25	
5	1	2	5	3.25	
9	3	3	5	5.00	
4	2	6	2	3.50	
7	7	1	6	5.25	
9	1	5	4	4.75	
5	3	3	9	5.00	
7	8	4	1	5.00	
0	5	6	1	3.00	
9	8	2	2	5.25	
6	1	5	7	4.75	
8	1	3	0	3.00	
5	9	6	9	7.25	
6	2	3	4	3.75	
7	4	0		4.50	
5	5	7	6	5.75	
5	2	8	6	5.00	
2	0	9	7	4.50	
5	8	9	0	5.50	
6	5	4	9	6.00	
4	8	7	6	6.25	
7	1	2	0	2.50	
2	9	5	0	4.00	
8	3	2	2	3.75	
2	7	1	6	4.00	
6	7	7	1	5.25	
2	3	3	9	4.25	
2	4	7	5	4.50	
5	4	3	7	4.75	
0	4	3	8	3.75	
2	5	8	6	5.25	
7	1	3	4	3.75	
8	3	7	0	4.50	
5	6	6	7	6.00	

# Testes de significância estatística - revisão

- Inferência estatística: métodos para tirar conclusões a partir de dados.
- Probabilidades expressam a *força* das conclusões;
- Testes de significância se baseiam em distribuições amostrais de estatísticas;
- Probabilidades para afirmar o que aconteceria se utilizássemos o método de inferência muitas vezes.
- Ter em mente a importância de se realizar *experimentos controlados*.

## **Objetivo de um Teste de Significância (TS):**

- Avaliar a evidência oferecida pelos dados em favor de uma afirmação sobre a população.

## **Raciocínio subjacente:**

- O que aconteceria se repetíssemos muitas vezes a amostra (experimento)?

## **Exemplo 1 (Moore, 2000):**

- Tomates - embalagens de 227g.



- Supor que uma amostra revele que o peso médio das embalagens é de 225g ( $< 227$ g na embalagem);
- Como não poderíamos esperar que todas as embalagens pesariam exatamente 227g:
  - A diferença se deve simplesmente *ao acaso*?

OU

- Máquina empacotadora está com problemas?

- Um TS testa uma hipótese específica, usando dados amostrais para decidir sobre sua validade.
- Em nosso exemplo, queremos saber se  **$\mu = 227 \text{ g}$** :
  - Hipótese nula ( $H_0$ ): é a afirmação sendo testada;
    - Proposição sobre ausência de diferença.
    - TS avaliará a força da evidência contra  $H_0$ .
- A hipótese alternativa ( $H_1$ ) é a afirmação para a qual procuramos evidência. Por exemplo,  **$H_1: \mu \neq 227 \text{ g}$** .
- A evidência que usaremos para decidir entre  $H_0$  e  $H_1$  é a média da nossa amostra de dados.

## **Lógica de um TS:**

- Assumir  $H_0$  verdadeira (embora ela possa ser falsa);
- Qual é a probabilidade de obter dados *tão extremos* quanto aqueles de que dispomos se  $H_0$  é verdadeira?
  - Improvável:
    - Tendemos a *duvidar* de  $H_0$ .
  - Provável:
    - Tendemos a *acreditar* em  $H_0$ .
- O que significa obter dados *tão extremos* quanto aqueles de que dispomos?

- Máquina de empacotamento necessita de revisão?  
 $H_0: \mu = 227g$      $X$      $H_1: \mu \neq 227g$
- Qual é a probabilidade de extrair uma amostra aleatória tal como a nossa se  $H_0$  é verdadeira?



Média amostral = 225g,  $n = 4$ ;  
Desvio padrão populacional conhecido = 5g.



Em termos mais precisos:

- Se assumimos  $H_0$  verdadeira, qual é a distribuição amostral para as médias das amostras de tamanho 4?

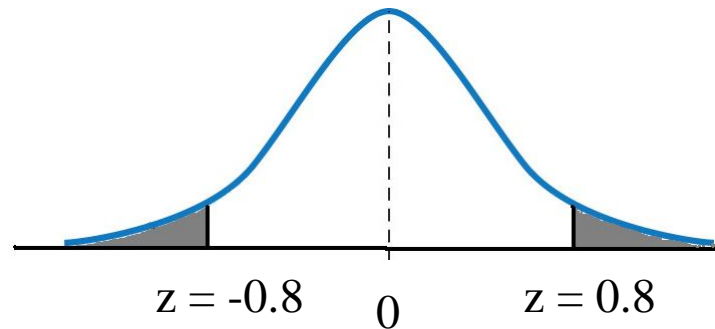
$$\bar{x} \sim N\left(\mu(\bar{x}) = 227, \sigma(\bar{x}) = \frac{5}{\sqrt{4}} = 2.5\right)$$

- Qual é o *escore*  $z$  da média amostral de que dispomos? Este valor é aqui denominado de ***estatística de teste***:

$$z = \frac{\bar{x} - \mu(\bar{x})}{\sigma(\bar{x})} = \frac{225 - 227}{2.5} = -0.8$$

**Obs.:** Estamos presumindo que  $X$  possui distribuição normal, mas essa restrição pode ser relaxada ...

- Quais são os outros escores  $z$  tão extremos quanto o disponível (na *direção* de  $H_1$ )?

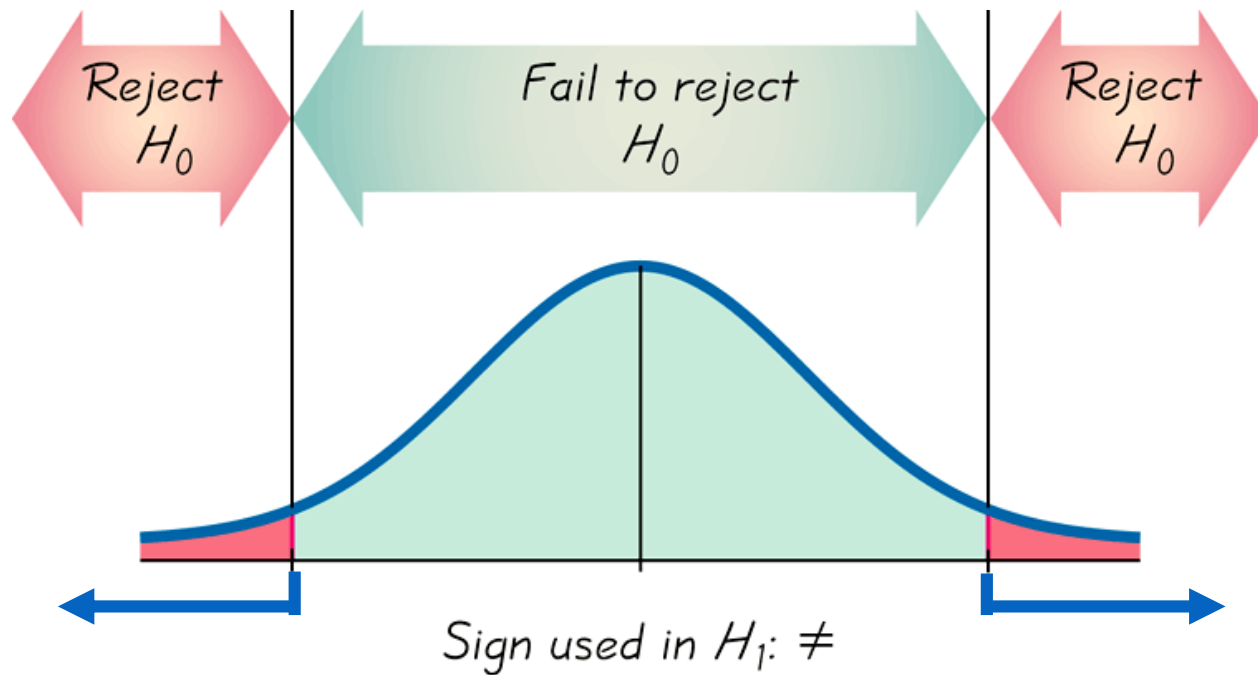


- Quão provável é obter uma média amostral tão extrema quanto a nossa?
  - Valor  $P$  do TS.
  - $P = 0,4237$ .

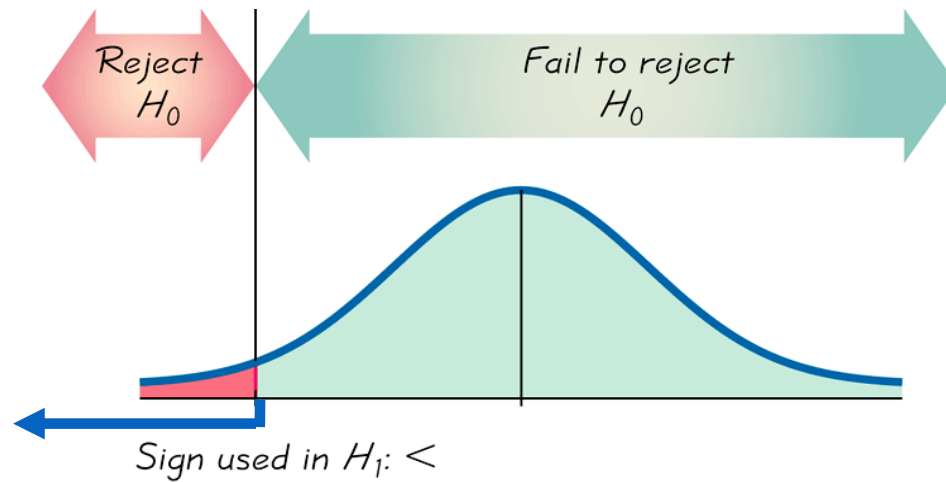
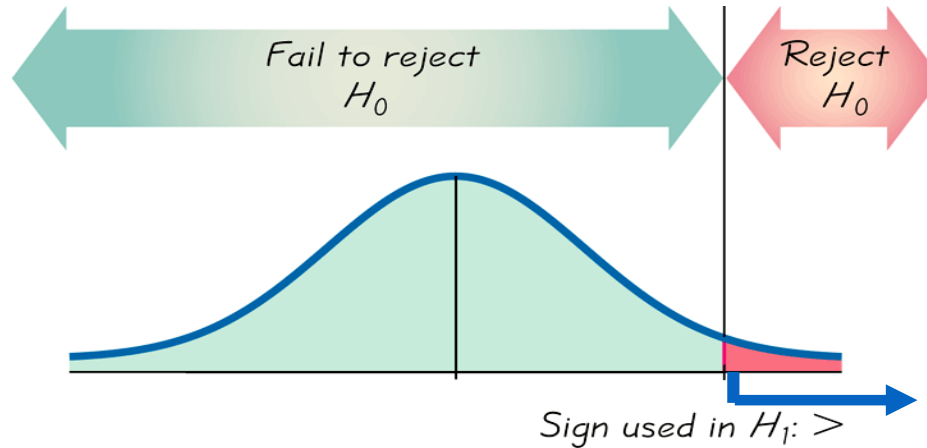
- $P$  é a probabilidade de que somente uma variação aleatória (do processo de amostragem) é responsável pela diferença observada.
- Valor pequeno de  $P$  implica que a variação aleatória provavelmente NÃO é a única responsável pela diferença observada;
- Rejeitar  $H_0$ , i.e., há evidência de que a verdadeira propriedade da população é significativamente diferente de  $H_0$ .
- O que pode ser considerado um valor pequeno de  $P$ ?

- Nível de significância ( $\alpha$ ) é o maior valor tolerado para  $P$  a fim de rejeitarmos  $H_0$ , refletindo quanta evidência necessitamos contra  $H_0$ :
  - Se  $P \leq \alpha$  : rejeitamos  $H_0$ .
  - Se  $P > \alpha$  : falhamos em rejeitar  $H_0$  .
- Valores comuns para  $\alpha$ : 10%, 5% e 1%.
- No nosso exemplo,  $P=42,37\%$ . O que concluimos?

- TS unilaterais (unicaudais) e bilateral (bicaudal):
- TS bilateral:  $H_0$  contém o sinal de igualdade (=).



- TS unilaterais direito e esquerdo:



# Diretrizes

- $n \geq 30$ : escore  $z$ , estimando  $\sigma=s$ ;
- $n < 30$  e histograma essencialmente *não normal*: métodos não paramétricos;
- $n < 30$  e histograma *normal*, com  $\sigma$  conhecido: usar escore  $z$ ;
- $n < 30$  e histograma *normal*, com  $\sigma$  desconhecido: usar estatística  $t$ ;

# Comparando dois algoritmos

## Cenários típicos:

- Reportar desempenho de um *novo* (desafiante) algoritmo;
  - Comparar diversos algoritmos num problema particular (e.g., estimativa de renda);
- Foco em alguma(s) medida(s) que capture(m) a capacidade do algoritmo em solucionar o problema.

## Testes usualmente empregados:

- Teste  $t$ ;
- Teste de Wilcoxon.
- Para ilustrar, veremos como comparar dois modelos;



- Presumir que  $c_i^j$  é um escore de desempenho do  $j$ -ésimo algoritmo na  $i$ -ésima base de dados.
  - Para os valores  $c_i^j$  obtidos: diferenças de desempenho são estatisticamente significativas?
- Preferivelmente os algoritmos devem ser rodados nas mesmas amostras: planejar bem os experimentos.

## Exemplo 2 (Demsar, 2006):

	Alg. A	Alg. B
Adult	0.763	0.768
Breast	0.599	0.591
Wisconsin	0.954	0.971
Cmc	0.628	0.661
Ionosphere	0.882	0.888
Iris	0.936	0.931
Liver	0.661	0.668
Lung	0.583	0.583
Lymph...	0.775	0.838
Mushroom	1.000	1.000
Tumor	0.940	0.962
Rheum	0.619	0.666
Voting	0.972	0.981
Wine	0.957	0.978

Demsar, J. Statistical Comparisons of Classifiers over Multiple Data Sets, *Journal of Machine Learning Research* 7, 1-30, 2006.

## Teste $t$ (revisão):

- Verificar se a diferença média (B-A) é significativamente diferente de zero;
- $n < 30$ , histograma *normal* (?),  $\sigma$  desconhecido.

## *Problemas:*

- Diferenças para as bases de dados são “comensuráveis”?
- Diferenças entre as 2 variáveis aleatórias são *Normais*?
- Médias afetadas por *outliers*?

	Alg. A	Alg. B
Adult	0.763	0.768
Breast	0.599	0.591
Wisconsin	0.954	0.971
Cmc	0.628	0.661
Ionosphere	0.882	0.888
Iris	0.936	0.931
Liver	0.661	0.668
Lung	0.583	0.583
Lymph...	0.775	0.838
Mushroom	1.000	1.000
Tumor	0.940	0.962
Rheum	0.619	0.666
Voting	0.972	0.981
Wine	0.957	0.978

- Computando a estatística de teste:

$$t = \frac{\bar{x} - \mu(\bar{x})}{s/\sqrt{n}} = \frac{0,0155 - 0,0000}{0,02/3,74} = 2,89$$

- $t_{crítico}$  para  $\alpha=1\%$  (unilateral) é 2,65.
- O que concluimos?

## Teste de Wilcoxon:

- Baseado nos *ranks* das diferenças;
- Assume comensurabilidade qualitativa (*ranks / postos*):
  - Grandes diferenças contam mais;
  - Magnitudes das diferenças não são levadas em conta.
- Não presume distribuições *Normais*;
- Efeito dos *outliers* é atenuado.

Voltemos ao *Exemplo 2...*

	Alg. A	Alg. B	Diferença	Rank/Posto
Adult	0.763	0.768	+0.005	3.5
Breast	0.599	0.591	-0.008	7
Wisconsin	0.954	0.971	+0.017	9
Cmc	0.628	0.661	+0.033	12
Ionosphere	0.882	0.888	+0.006	5
Iris	0.936	0.931	-0.005	3.5
Liver	0.661	0.668	+0.007	6
Lung	0.583	0.583	0.000	1.5
Lymph...	0.775	0.838	+0.063	14
Mushroom	1.000	1.000	0.000	1.5
Tumor	0.940	0.962	+0.022	11
Rheum	0.619	0.666	+0.047	13
Voting	0.972	0.981	+0.009	8
Wine	0.957	0.978	+0.021	10

Analizando  
intuitivamente o  
sumário obtido...

	(B – A)	<i>Rank</i>
Adult	+0.005	3.5
Breast	-0.008	7
Wisconsin	+0.017	9
Cmc	+0.033	12
Ionosphere	+0.006	5
Iris	-0.005	3.5
Liver	+0.007	6
Lung	0.000	1.5
Lymph...	+0.063	14
Mushroom	0.000	1.5
Tumor	+0.022	11
Rheum	+0.047	13
Voting	+0.009	8
Wine	+0.021	10

Ranks favoráveis ao Algoritmo B:

$$R^+ = 3.5 + 9 + 12 + 5 + 6 + 14 + 11 + 13 + 8 + 10 + 1.5 = 93$$

Ranks favoráveis ao Algoritmo A:

$$R^- = 7 + 3.5 + 1.5 = 12.$$

Dados sugerem que *B* é melhor do que *A*...

→ Há significância estatística?

Estatística de teste:

$$T = \min(R^+, R^-)$$

→ Valores críticos para *T*

disponíveis em pacotes/tabelas;


→ Neste caso, rejeita-se  $H_0$  para  $\alpha=5\%$  ( $T_{\max}=17$ ).

	(B – A)	Rank
Adult	+0.005	3.5
Breast	-0.008	7
Wisconsin	+0.017	9
Cmc	+0.033	12
Ionosphere	+0.006	5
Iris	-0.005	3.5
Liver	+0.007	6
Lung	0.000	1.5
Lymph...	+0.063	14
Mushroom	0.000	1.5
Tumor	+0.022	11
Rheum	+0.047	13
Voting	+0.009	8
Wine	+0.021	10



# Observações finais

- Muito debate sobre o uso de TS;
- Vimos duas abordagens razoavelmente bem aceitas;
- Resultados de TS devem ser usados com sobriedade;
- TS proporcionam alguma evidência da validade dos resultados obtidos;
- Significância estatística X significância prática?



The screenshot shows the top portion of a web page from the journal Nature. The header is dark red with the 'nature' logo in white. Below the logo, it says 'International weekly journal of science'. A navigation bar contains links: Home, News & Comment, Research, Careers & Jobs, Current Issue, Archive, and Audio & Video. Below this, a breadcrumb trail shows: Archive > Volume 506 > Issue 7487 > News Feature > Article. The main content area has a sub-header 'NATURE | NEWS FEATURE' and a language selector 'عربي'. The article title is 'Scientific method: Statistical errors'. The lead text reads: 'P values, the 'gold standard' of statistical validity, are not as reliable as many scientists assume.' The author's name, 'Regina Nuzzo', is listed below. The date '12 February 2014' is at the bottom of the article preview.

**nature** International weekly journal of science

Home | News & Comment | Research | Careers & Jobs | Current Issue | Archive | Audio & Video

Archive > Volume 506 > Issue 7487 > News Feature > Article

NATURE | NEWS FEATURE

عربي

## Scientific method: Statistical errors

P values, the 'gold standard' of statistical validity, are not as reliable as many scientists assume.

Regina Nuzzo

12 February 2014

# Agenda

- Árvores
- Random Forests
- k-Nearest Neighbors
- LASSO (Least Absolute Shrinkage and Selection Operator)
- Avaliação de Modelos