

 II WORKSHOP EM INTELIGÊNCIA COMPUTACIONAL E
APRENDIZADO ESTATÍSTICO APLICADOS À AGROPECUÁRIA
10 A 13 DE SETEMBRO

MINICURSOS

MINICURSO 1: REDES NEURAIS UTILIZANDO O SOFTWARE GENES

MINISTRANTE: PROF. COSME DAMIÃO CRUZ -DEPARTAMENTO DE BIOLOGIA
GERAL /UFV

DATA: 10 DE SETEMBRO DE 2019

HORÁRIO: 8 :00 - 12:00

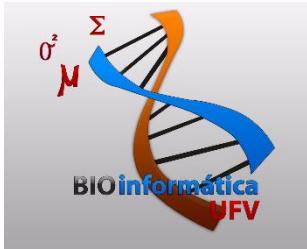
LOCAL: LABORATÓRIO DE BIOINFORMÁTICA- BIOAGRO/UFV- SALA 109



Programa do Curso:

1. Inteligência Computacional - Histórico
2. Aplicação 1: Classificação
3. Aplicação 2: Predição

Equipe:



Laboratório de Bioinformática

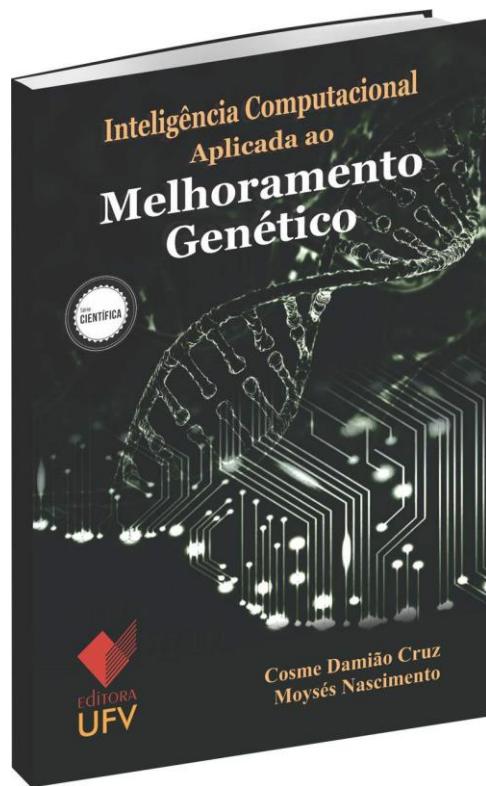


Laboratório de Inteligência
Computacional e Aprendizado
Estatístico



Laboratório de Biometria

REDES NEURAIS: CONTEXTUALIZAÇÃO



REDES NEURAIS

1. Considerações iniciais

Inteligência Computacional

Área da ciência que estuda a teoria e a aplicação de técnicas inspiradas na Natureza, como:

Redes Neurais

Lógica Nebulosa

Computação Evolucionária.

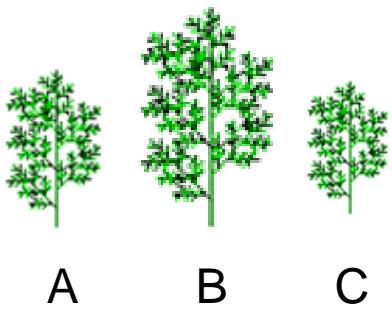
Redes Neurais

Redes Neurais são modelos computacionais não lineares, inspirados na estrutura e operação do cérebro humano, que procuram reproduzir características humanas, tais como:

aprendizado, associação, generalização e abstração.

Devido à sua estrutura, as Redes Neurais são bastante efetivas no aprendizado de padrões a partir de dados não-lineares, incompletos, com ruído e até compostos de exemplos contraditórios.

2. Como podemos decidir qual é o melhor genótipo?



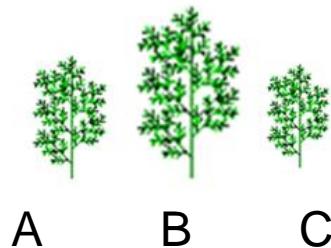
Resposta por uma
abordagem estatística

Resposta por uma
abordagem de inteligência
computacional



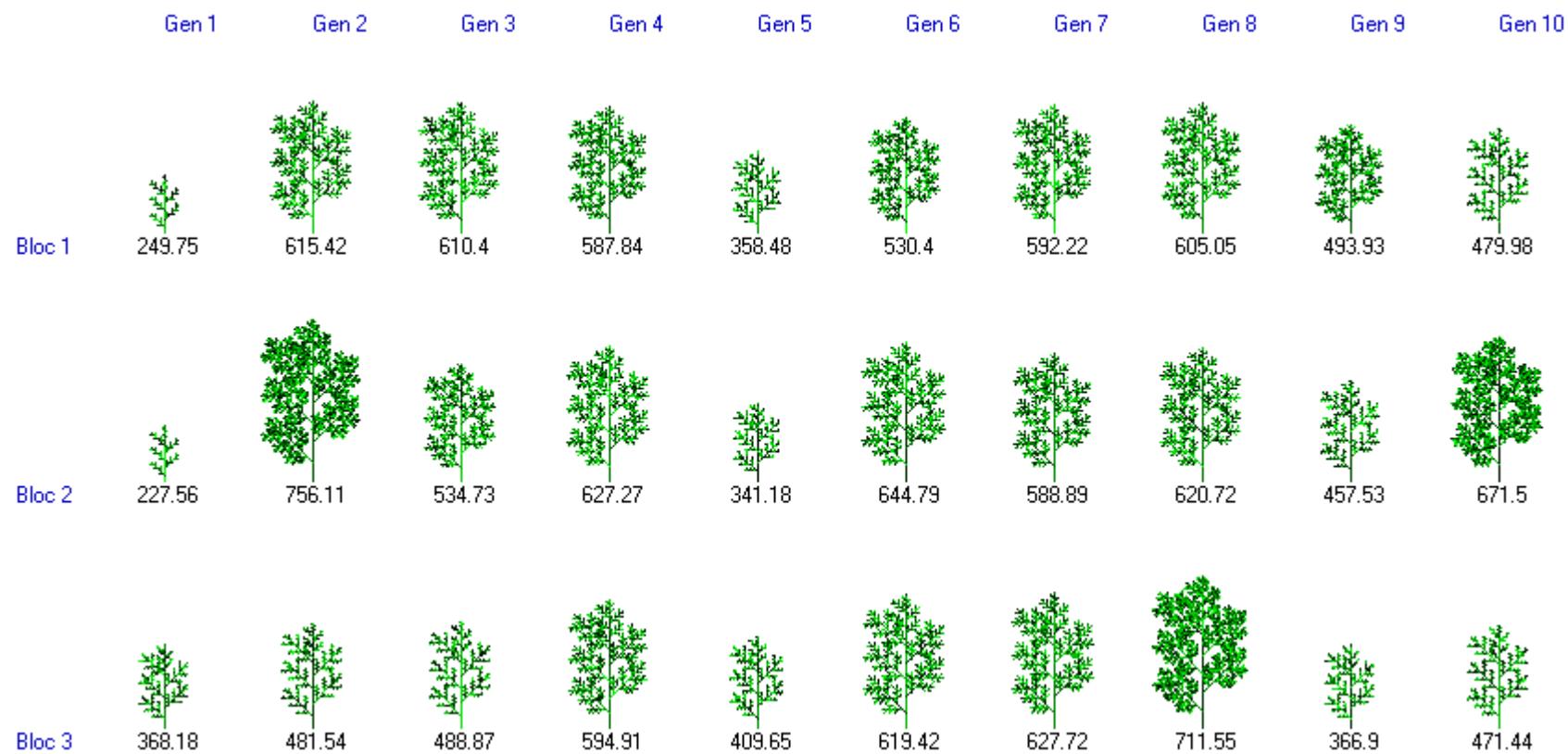
3. Tomada de decisão

3.1 Tomada de decisão baseada em modelo estatístico

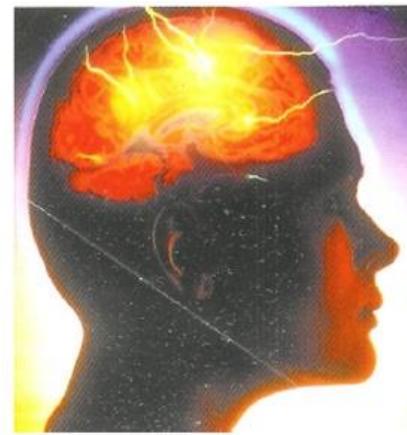


$$Y_{ij} = m + G_i + B_j + E_{ij}$$

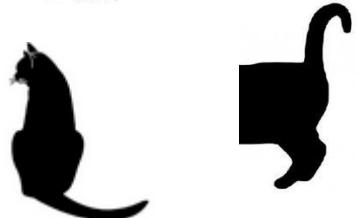
- ❖ Modelo estatístico
- ❖ Pressuposições e distribuições
- ❖ Análises estatísticas



3.2. Tomada de decisão baseada em modelo biológico



Que animal é este ???

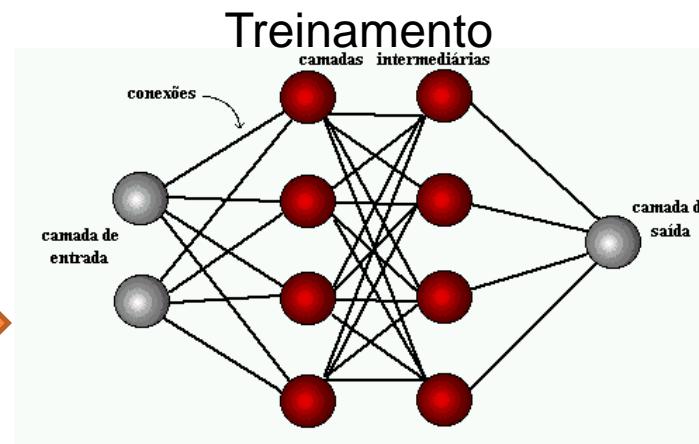
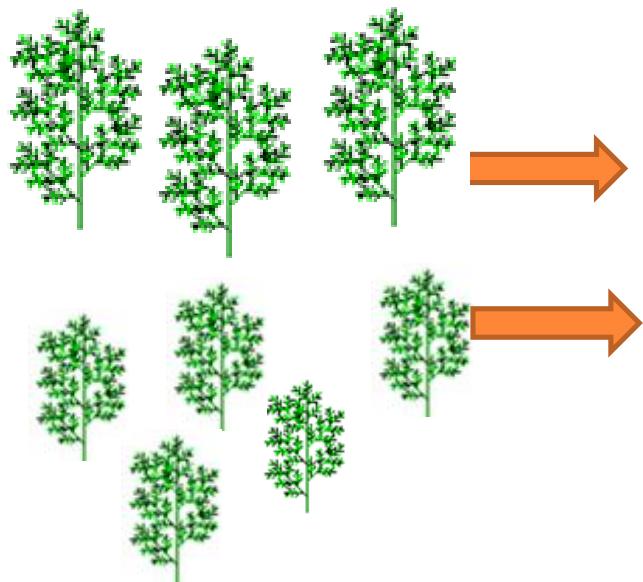


Treinamento e aprendizagem

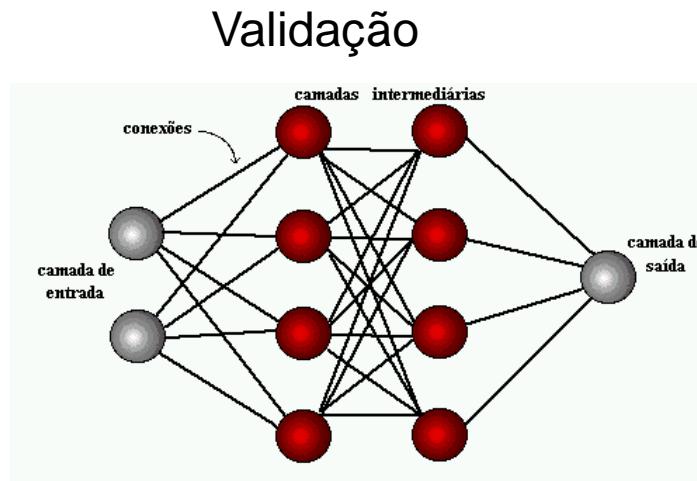
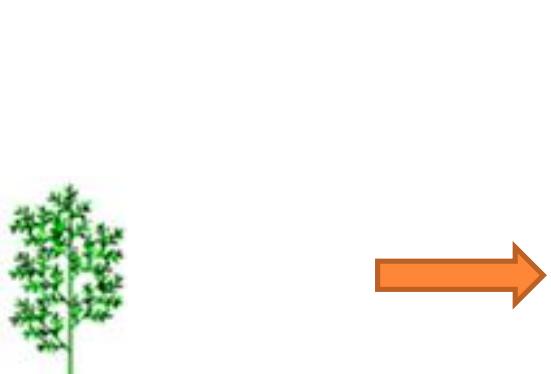
{ G - N - T - C -
G - - R - C - M -



3.3. Tomada de decisão na inteligência computacional



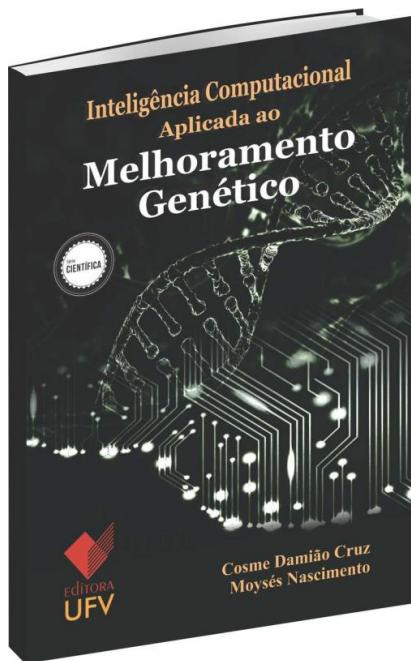
Boa
Ruim



?



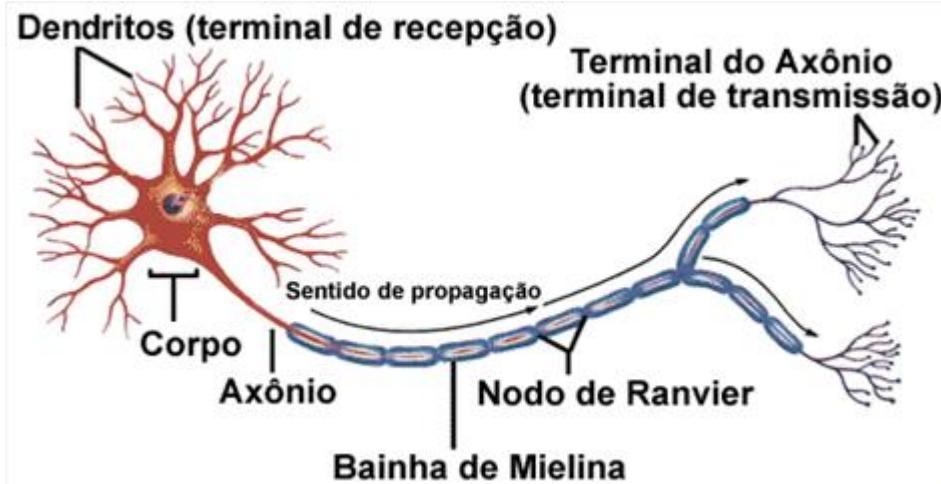
REDES NEURAIS: FUNDAMENTOS



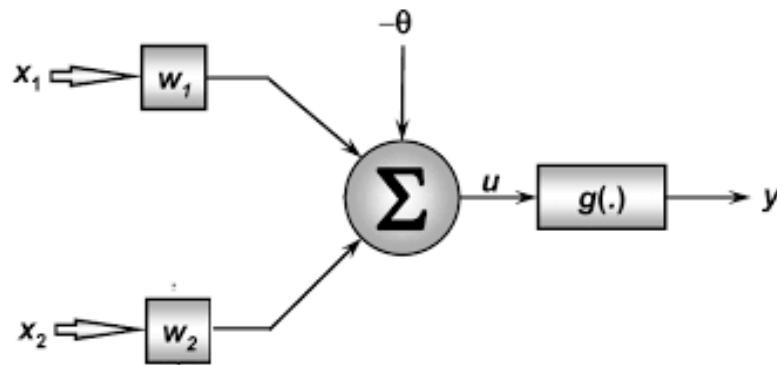
1. Introdução

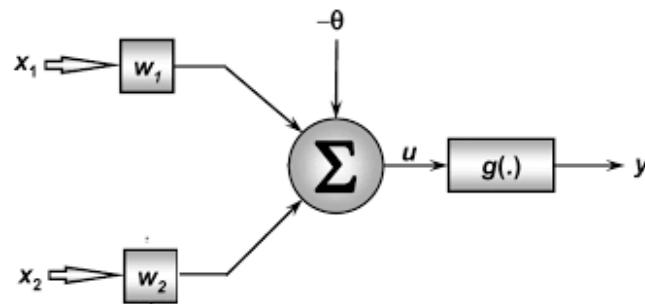
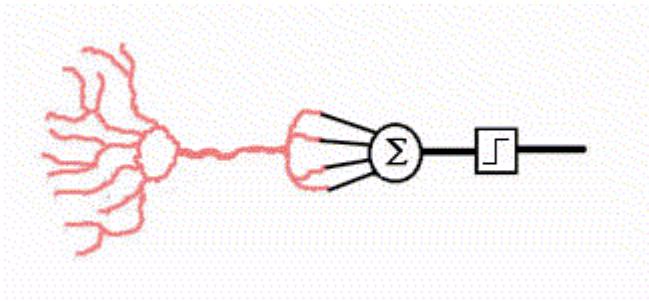
As abordagens por RNA tentam imitar os princípios de um neurônio biológico

Neurônio biológico



Topologia de rede





2. Modelo de McCulloch e Pitts



Warren McCulloch



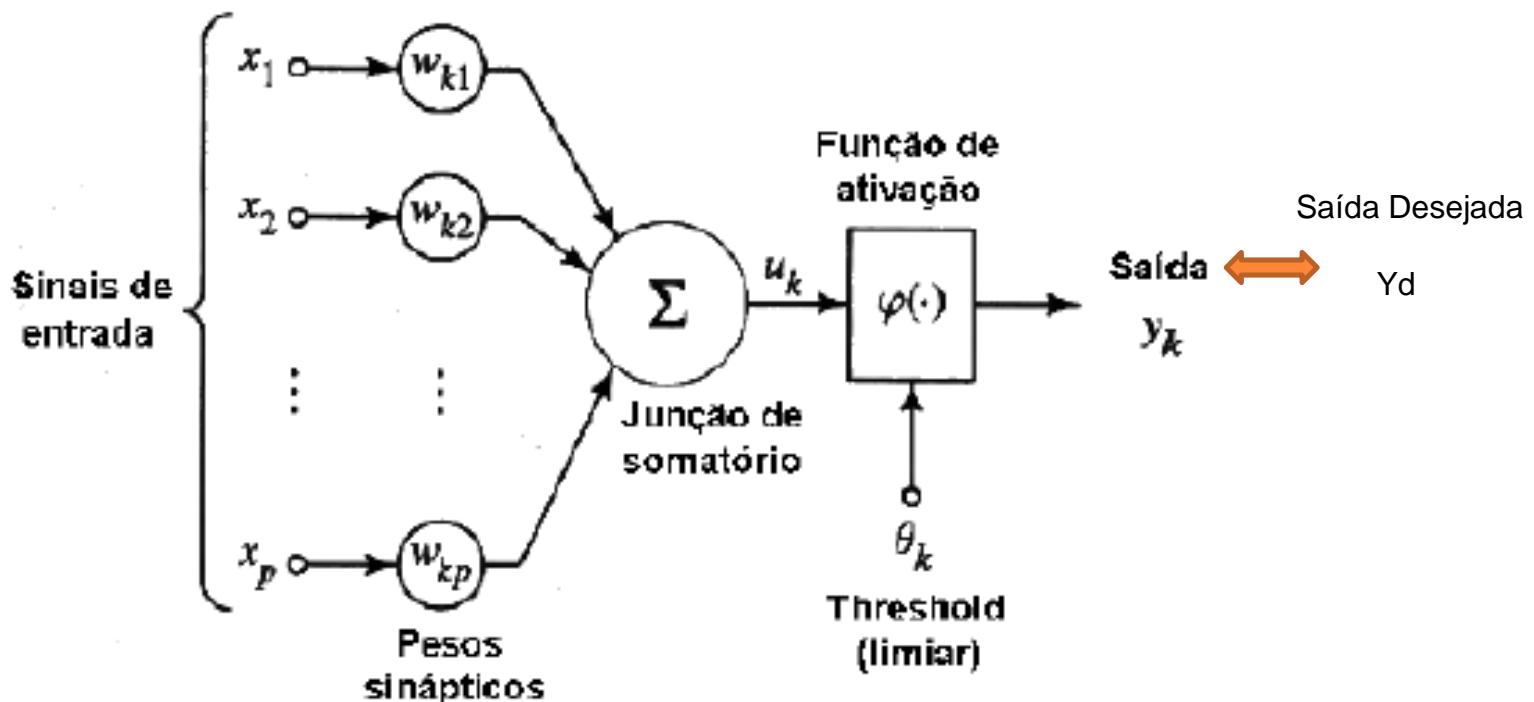
Walter Pitts

McCulloch era um psiquiatra e um neuroanatomista e Pitts era um matemático que foi trabalhar com McCulloch na Universidade de Chicago, ambos fazendo parte de um dos primeiros grupos do mundo dedicado ao estudo da Biofísica Teórica, criado por Nicolas Rashevsky.

Em 1943, o conhecimento sobre os neurônios biológicos era muito limitado. As bases iônicas e elétricas da atividade neuronal eram ainda incertas, porém já se sabia da existência de potenciais de ação e da sua natureza “tudo-ou-nada”.

McCulloch e Pitts propuseram um modelo de sistema neural em que as unidades básicas, os neurônios, são bastante simples no seu funcionamento.

2.1 Neurônio de McCulloch-Pitts



A saída y do neurônio de McCulloch-Pitts pode ser equacionada por:

$$y = \varphi \left(\sum_{i=1}^n x_i w_i + \theta_k \right)$$

Annotations for the equation components:

- Saída da Rede (Output of the Network) points to y .
- Função de ativação (Activation Function) points to φ .
- Porta do Limiar (Threshold Gate) points to the summation term $\sum_{i=1}^n x_i w_i + \theta_k$.
- Limiar (Threshold) points to θ_k .

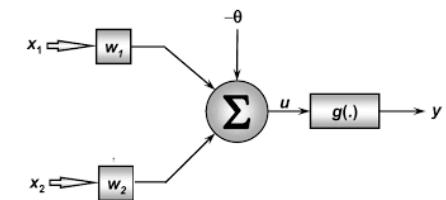
2.3. Solução por meio de RNA - Ilustração

(w_1) x_1	(w_2) x_2	$\sum w_i x_i$	$\theta = 16$ $Y_r = 1 \text{ se } f(\sum w_i x_i < \theta)$ $Y_r = 2 \text{ se } f(\sum w_i x_i > \theta)$	Y_d	Erro = $Y_r - Y_d$
122.77	106.29	S1 = 12	1	1	0
110.01	111.08	S2 = 10	1	1	0
121.87	104.21	S3	1	1	0
125.21	114.94	S4	1	1	0
124.02	110.23	S5	1	1	0
125.52	105.07	S6	1	1	0
117.28	106.8	S7	1	1	0
112.16	112.43	S8	1	1	0
123.35	118.12	S9	1	1	0
117.76	110.81	S10 = 15	1	1	0
138.55	120.98	S11 = 22	2	2	0
148.3	111.9	S12 = 25	2	2	0
139.58	129.88	S13	2	2	0
132.88	110.37	S14	2	2	0
133.51	121.71	S15	2	2	0
136.33	112.81	S16	2	2	0
134.78	124.79	S17	2	2	0
145.28	138.13	S18	2	2	0
154.31	114.1	S19	2	2	0
136.43	115.29	S20 = 28	2	2	0

Conceitos

- Pesos
- Somatório (porta do limiar)
- Limiar
- Função de ativação
- Saída de rede
- Saída desejada
- Erro – EQ

Aprendizado
EQ=0



2.4 Aplicação

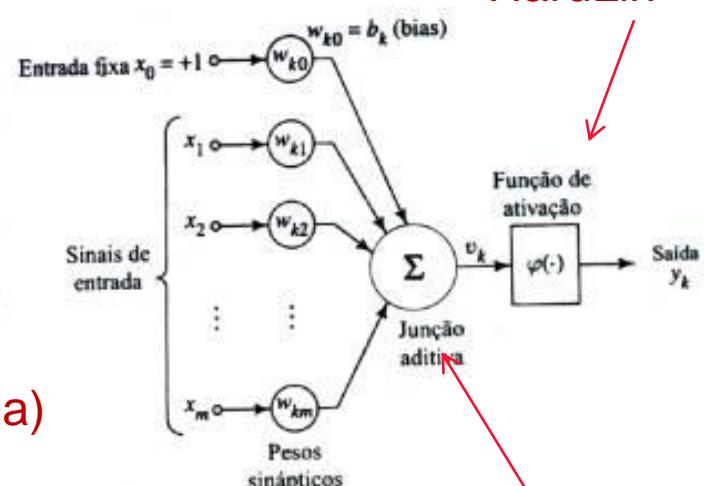
A tabela a seguir apresenta alguns exemplos de tomada de decisão que foram feitos por um especialista em análise de crédito. Crie um neurônio MCP para realizar estas classificações e, em seguida, faça a análise de crédito de um possível cliente.

Entrada (real)

Saída desejada(binária)

Tabela 1: Análise de Crédito

Idade	Sexo	Casa	Carro	Casado(a)	Nº filhos	Renda	Resultado
18	M	N	S	N	0	1200	N
19	M	S	S	S	1	700	S
25	F	N	N	S	2	800	S
40	M	S	N	S	4	800	N
21	M	N	N	N	0	1100	S
22	F	S	S	S	2	500	indefinido



HardLin

Porta de Limiar

Saída da rede

Rede
0
1
1
0
1

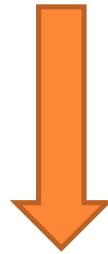
22	0	1	1	1	2	500	=	Sim
-.35	.43	-.175	-.361	.462	-.271	-.072	Bias =	-.267



Pesos

Limiar

Cliente	Idade	Sexo	Casa	Carro	Casado(a)	Filhos	Renda	Recomendação
1	18	M	N	S	N	0	1200	N
2	19	M	S	S	S	1	700	S
3	25	F	N	N	S	2	800	S
4	40	M	S	N	S	4	800	N
5	21	M	N	N	N	0	1100	S



$$X_c = (X_o - \min) / (\max - \min)$$

X_c e X_o : valores corrigido e original, respectivamente

max e min: valores máximo e mínimo da variável, respectivamente

Cliente	Idade	Sexo	Casa	Carro	Casado(a)	Filhos	Renda	Recomendação
1	0,0	0,0	0,0	1,0	0,0	0,0	1,0	0
2	0,0455	0,0	1,0	1,0	1,0	0,25	0,0	1
3	0,3182	1,0	0,0	0,0	1,0	0,5	0,2	1
4	1,0	0,0	1,0	0,0	1,0	1,0	0,2	0
5	0,1364	0,0	0,0	0,0	0,0	0,0	0,8	1



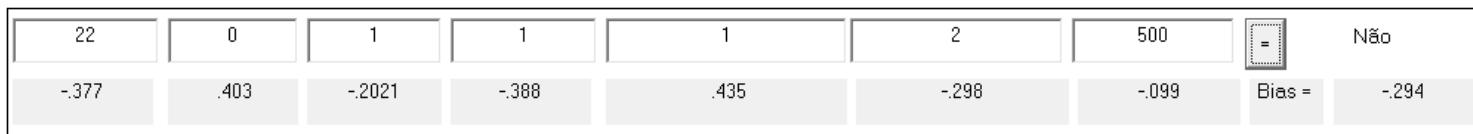
2.5. Soluções/Tarefa

Tabela 1: Análise de Crédito

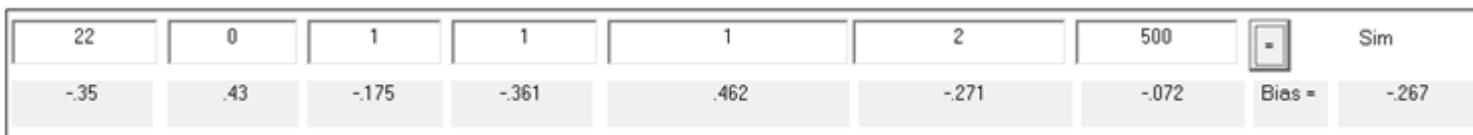
Idade	Sexo	Casa	Carro	Casado(a)	Nº filhos	Renda	Resultado
18	M	N	S	N	0	1200	N
19	M	S	S	S	1	700	S
25	F	N	N	S	2	800	S
40	M	S	N	S	4	800	N
21	M	N	N	N	0	1100	S
22	F	S	S	S	2	500	indefinido

Rede
0
1
1
0
1

Solução 1



Solução 2



mcp2.m

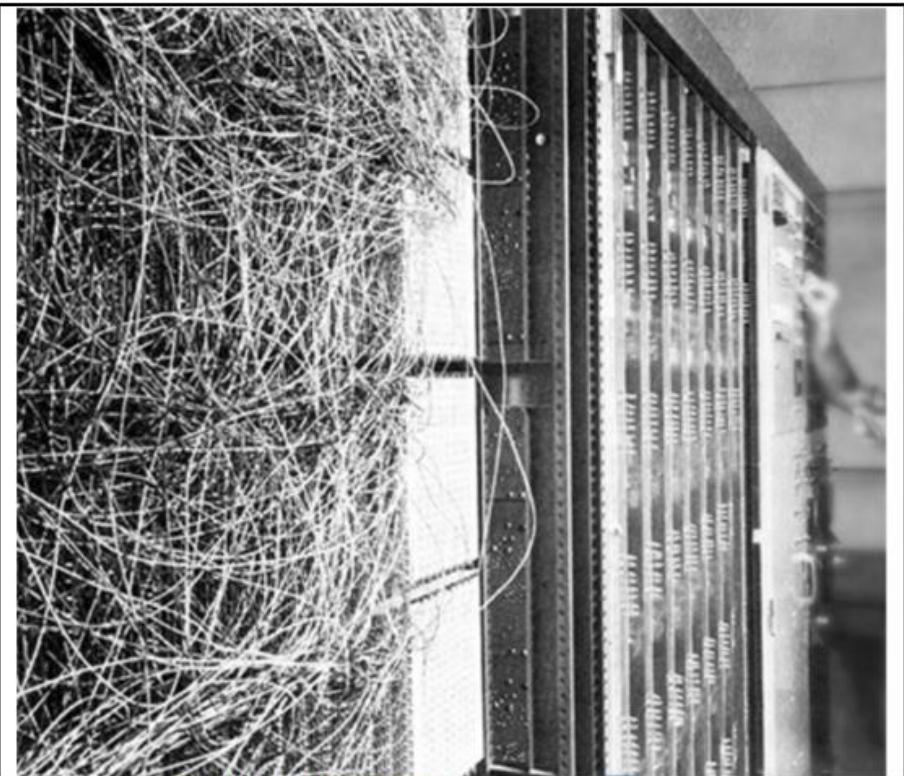


3. MODELO Perceptron

No final da década de 1950, Rosenblatt na Universidade de Cornell, criou uma genuína rede de múltiplos neurônios do tipo *discriminadores lineares* e chamou esta rede de *perceptron*.



Frank Rosenblatt

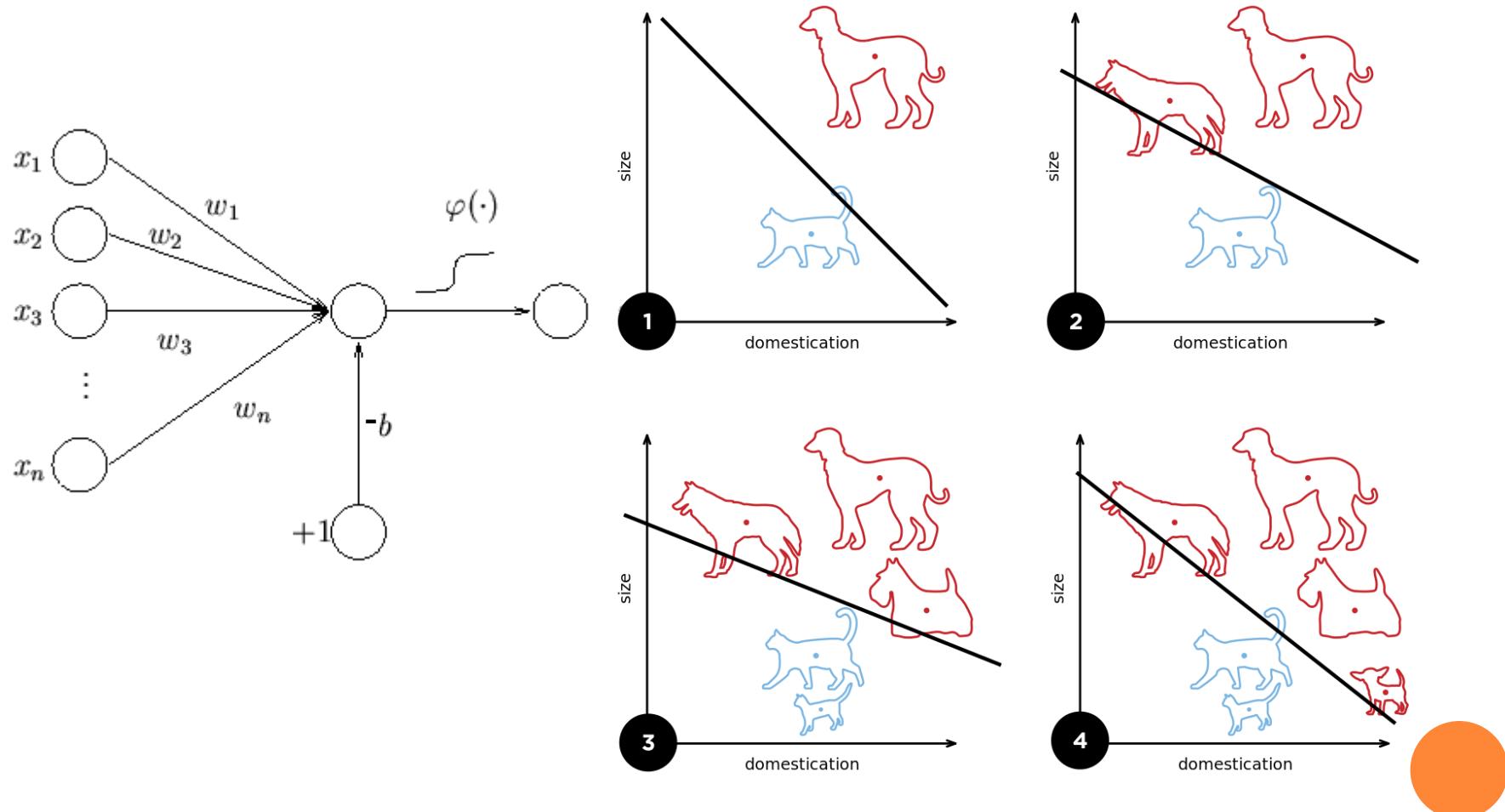


The Mark I Perceptron

Perceptron pode ser visto como o tipo mais simples de rede neural feedforward: um classificador linear.

3.1. Rede perceptron

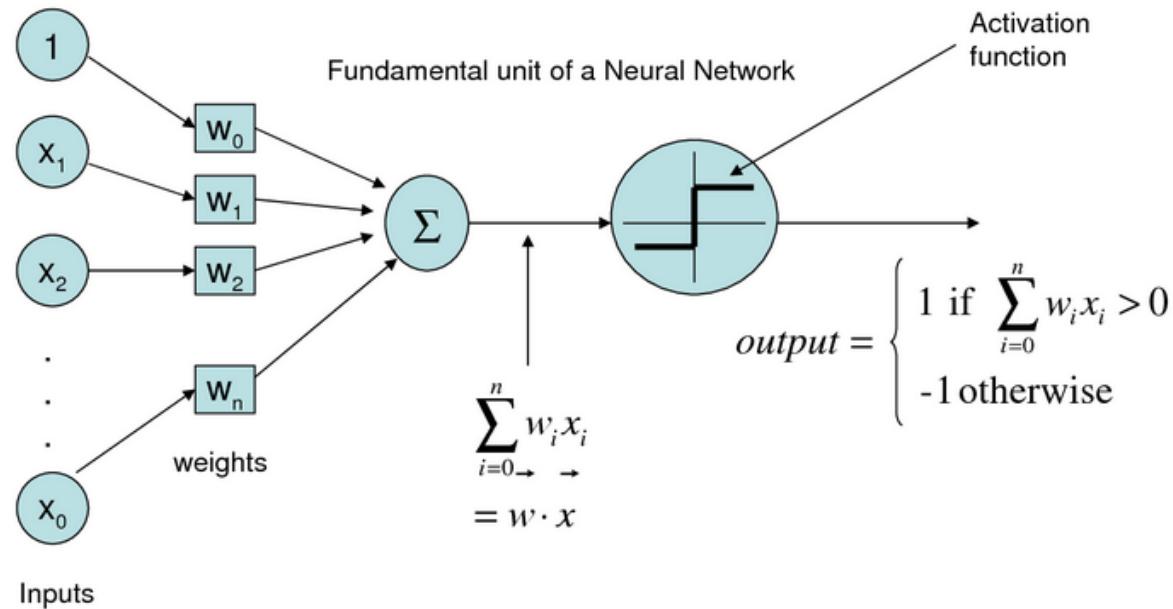
O Perceptron, proposto por Rosenblatt, é composto pelo neurônio de McCulloch-Pitts, com Função de Limiar, e [Aprendizado Supervisionado](#). Sua arquitetura consiste na entrada e uma camada de saída.



Existe adaptação para o perceptron

3.2 Regra Hebb

i. Introdução



Considerações:

$$\vec{x} = [1, x_1, x_2, \dots, x_n]$$

$$\vec{w} = [-b, w_1, w_2, \dots, w_n]$$

$$\sum_i w_i x_i = \vec{w} \cdot \vec{x}$$

produto
interno

$$\vec{w}(t+1) = \vec{w}(t) + ? \quad \Rightarrow \text{Regra Hebb}$$

Vetores de entrada e de pesos

$$\vec{x} = [1, x_1, x_2, \dots, x_n]$$

$$\vec{w} = [-\theta, w_1, w_2, \dots, w_n]$$

Função do limiar

$$f(w) = \sum_{i=0}^p w_i x_i = \vec{w} \vec{x}$$

Condições de disparo

$$\sum_{i=0}^p w_i x_i = \vec{w} \vec{x} < 0 \quad Y_r=0$$

$$\sum_{i=0}^p w_i x_i = \vec{w} \vec{x} > 0 \quad Y_r=1$$

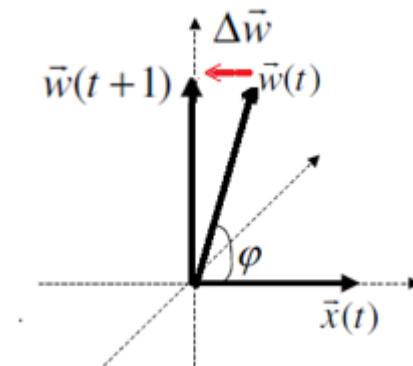
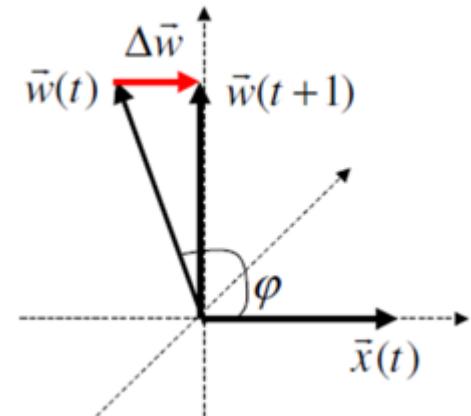
Aprendizado

Requer a obtenção de um vetor $\Delta \vec{w}$ a ser aplicado ao vetor de pesos \vec{w} ,

$$\vec{w}(t+1) = \vec{w}(t) + \Delta \vec{w}$$

Erro

$$\varepsilon = y - y_r \quad \varepsilon \neq 0: \begin{cases} y = 1, y_r = 0 \\ y = 0, y_r = 1 \end{cases}$$



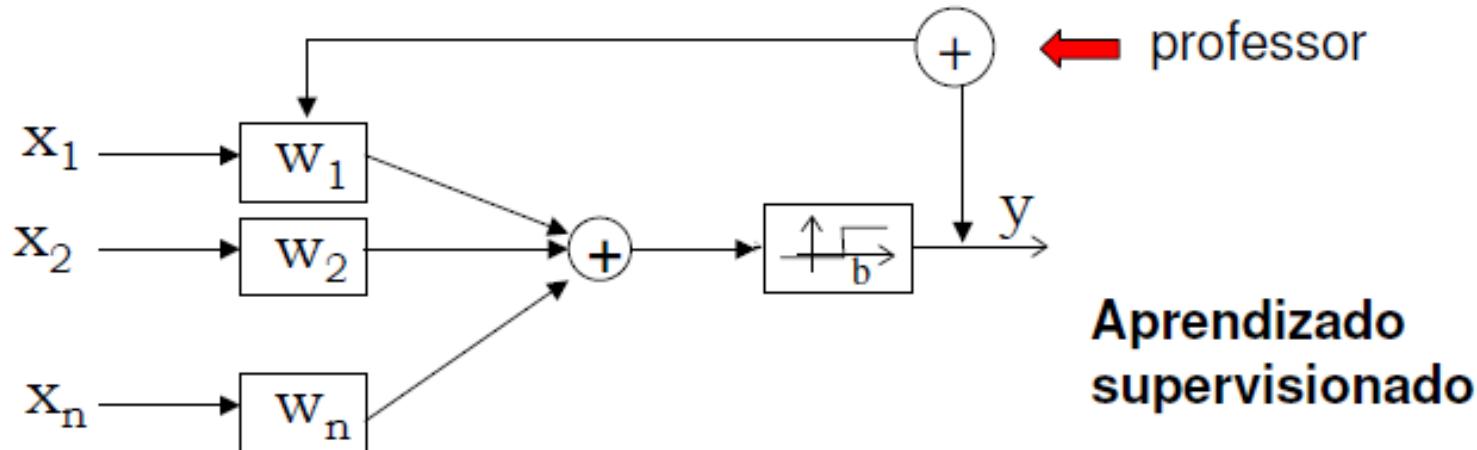
Nestas situações o vetor de pesos deve ser corrigido

Todavia, esse incremento só deve ser efetuado quando $e \neq 0$.

Ou seja, $\Delta \vec{w} = \eta \cdot e \cdot \vec{x}$.

Assim, finalmente:

$$\vec{w}(t+1) = \vec{w}(t) + \eta \cdot e \cdot \vec{x}$$



3.3. Exemplo

Nome	Febre	Enjoo	Manchas	Dores	Diagnóstico
João	S	S	Peq	S	Doente
Pedro	N	N	Grd	N	Saudável
Maria	S	S	Peq	N	Saudável
José	S	N	Grd	S	Doente
Ana	S	N	Peq	S	Saudável
Leila	N	N	Grd	S	Doente



Febre (X1)	Enjoo(X2)	Manchas(X3)	Dores(X4)	Diagnóstico(Yd)
1	1	0	1	0
0	0	1	0	1
1	1	0	0	1
1	0	1	1	0
1	0	0	1	1
0	0	1	1	0

Perceptron2



4. Modelo Adaline

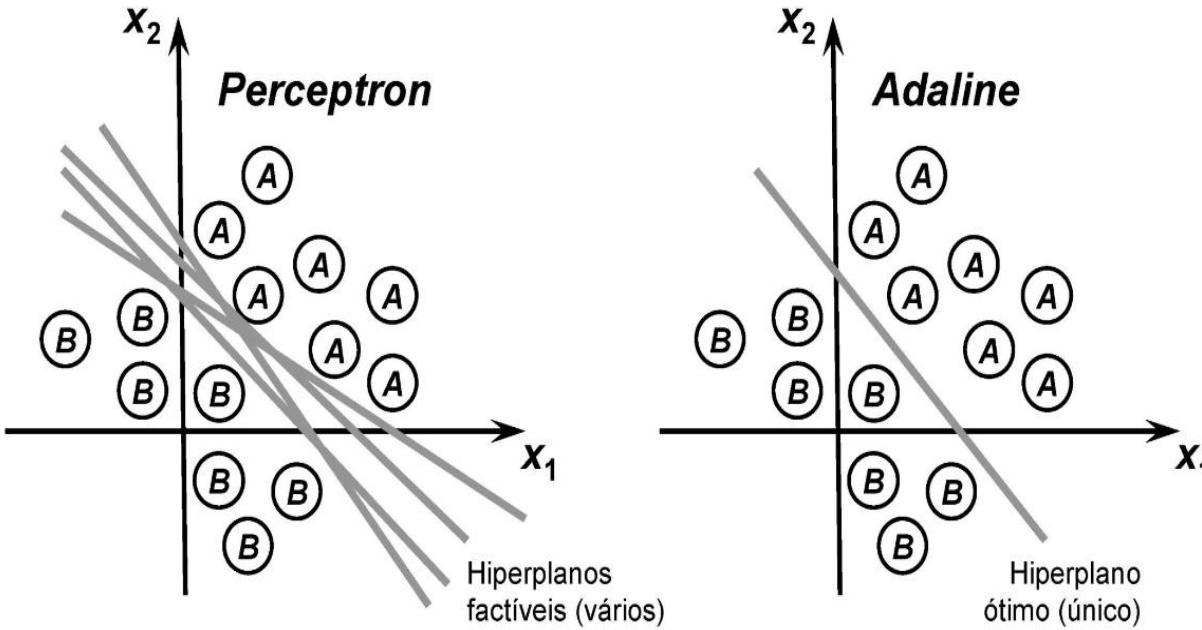
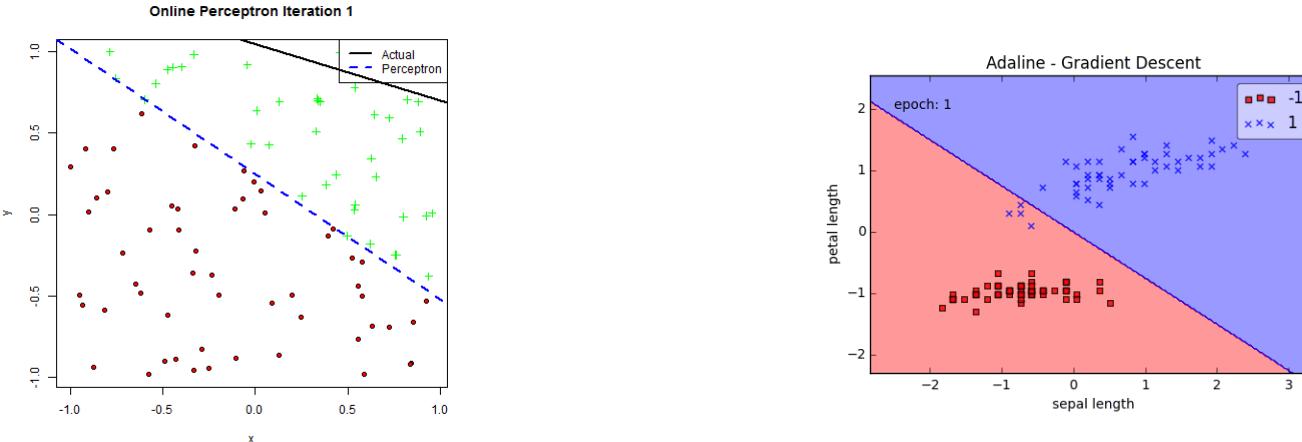
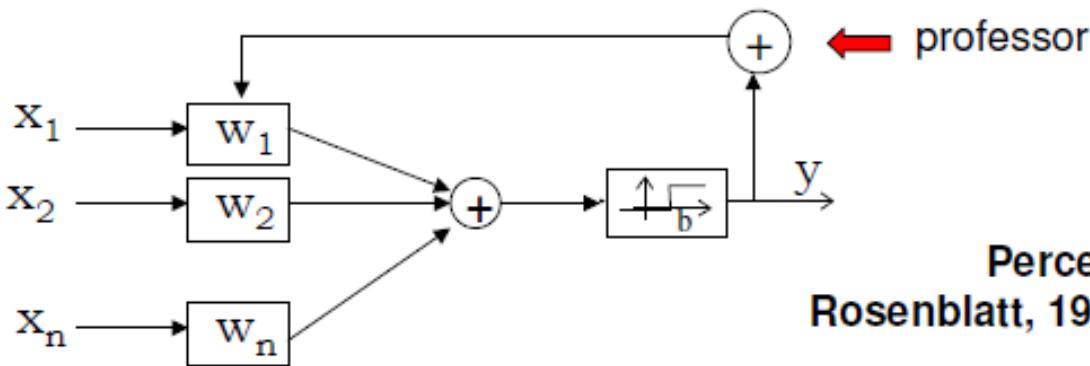


Figura 1. Possíveis hiperplanos de separação de duas populações A e B utilizando rede neural com arquitetura Perceptron e Adaline

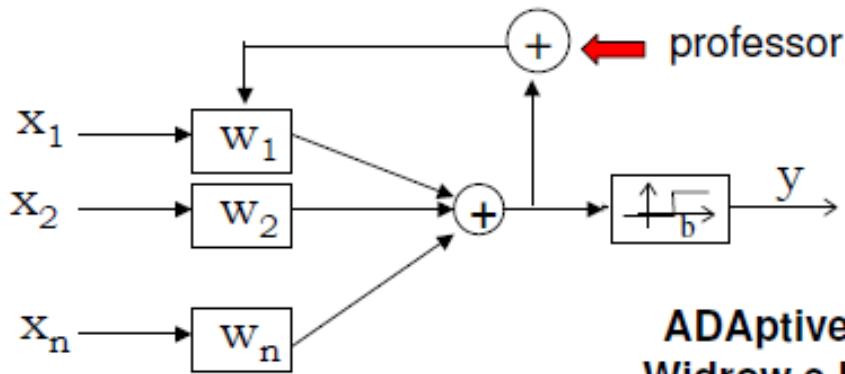


4.1. Adaline vs Perceptron

$$\text{Erro} = y_d - y_r$$



Perceptron
Rosenblatt, 1958 (psicologia)

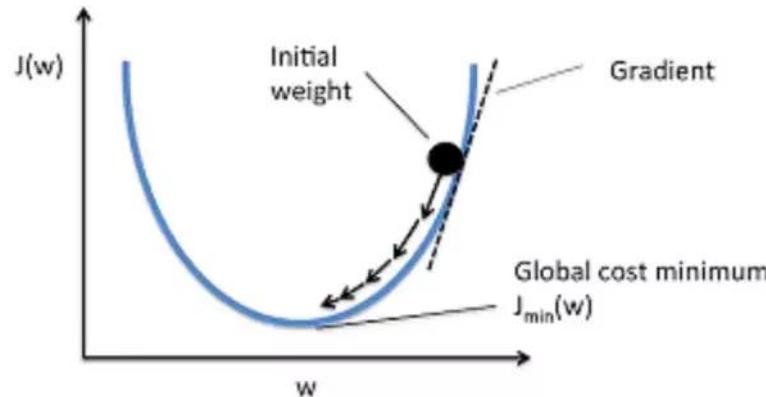
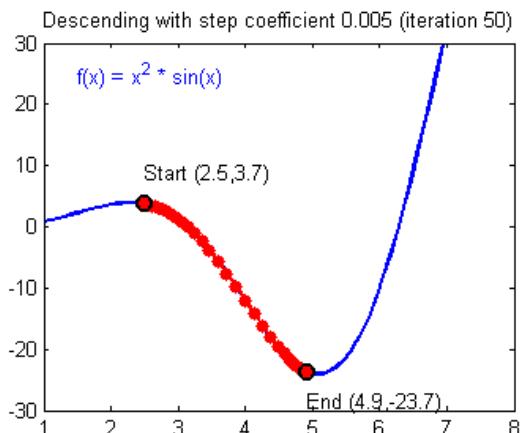


ADAptive LInear NEuron
Widrow e Hoff, 1960 (IEEE)

$$\text{Erro} = y_d - u$$

$$u = \text{potencial de ativação} = \sum w_i x_i - \theta$$

Obtenção de pontos de máximo ou mínimos pelo gradiente descendente



Usando o método do gradiente descendente, temos uma solução iterativa, em que os valores de x se aproximam a uma determinada taxa η por meio das expressões:

$$\Delta x(t) = -\eta \frac{\delta y}{\delta x} \quad \text{e} \quad x(t + 1) = x(t) + \Delta x(t)$$

Função de custo

$$E = \frac{1}{2} \sum_{j=1}^n (y_{dj} - \vec{w}\vec{x}_j)^2 = \frac{1}{2} [(y_{d1} - \vec{w}\vec{x}_1)^2 + (y_{d2} - \vec{w}\vec{x}_2)^2 + \dots + (y_{dn} - \vec{w}\vec{x}_n)^2]$$

Derivada

$$\frac{\partial E}{\partial \vec{w}} = \frac{1}{2} [2(y_{d1} - \vec{w}\vec{x}_1)(-\vec{x}_1) + 2(y_{d2} - \vec{w}\vec{x}_2)(-\vec{x}_2) + \dots + 2(y_{dn} - \vec{w}\vec{x}_n)(-\vec{x}_n)]$$

Assim, para uma dada observação, tem-se:

$$\frac{\partial E}{\partial \vec{w}_i} = -(y_{d1} - \vec{w}\vec{x}_i)(\vec{x}_i)$$

Erro 

$$= -\varepsilon_i \vec{x}_i$$

Aplicando a teoria do descendente tem-se:

$$\Delta w_i = \eta \sum \varepsilon_j x_{ij}$$

Regra de aprendizagem – Regra delta

$$w_{i(t+1)} = w_{i(t)} + \eta \sum \varepsilon_j x_{ij}$$



Aplicação – Classificação adaline2.m

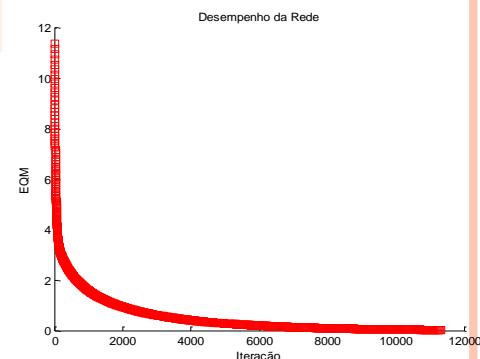
Tabela . Informações relativas ao histórico de seis pacientes em relação a ocorrência de determinada doença. S: sim; N: não; Peq: pequenas; Grd: grandes.

Paciente	Febre	Enjoo	Manchas	Dores	Diagnóstico
João	S	S	Peq	S	Doente
Pedro	N	N	Grd	N	Saudável
Maria	S	S	Peq	N	Saudável
José	S	N	Grd	S	Doente
Ana	S	N	Peq	S	Saudável
Leila	N	N	Grd	S	Doente



Resultados obtidos por análise, em cinco diferentes processamentos, utilizando rede Adaline

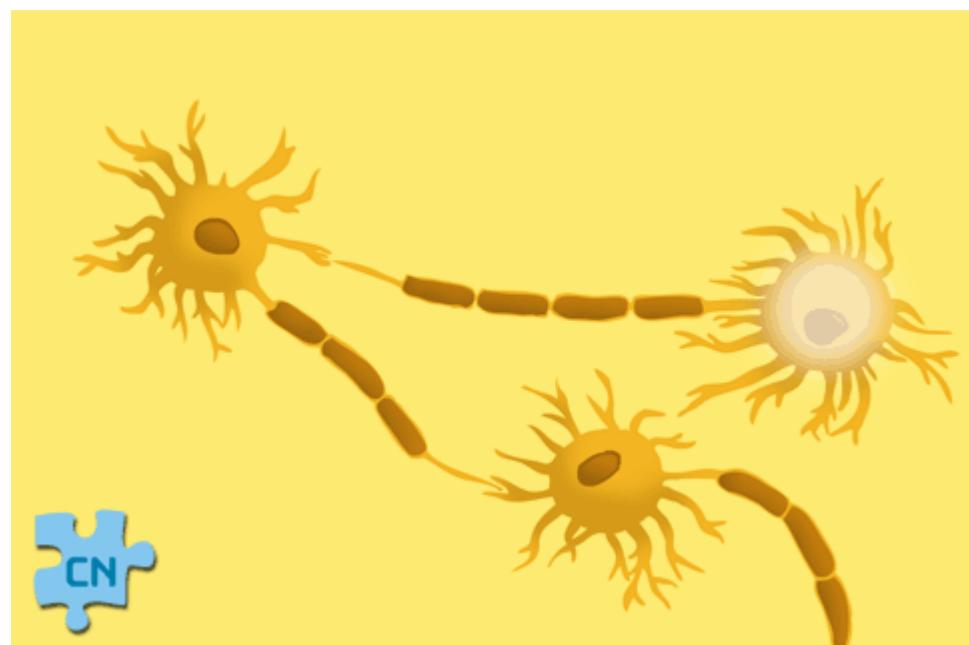
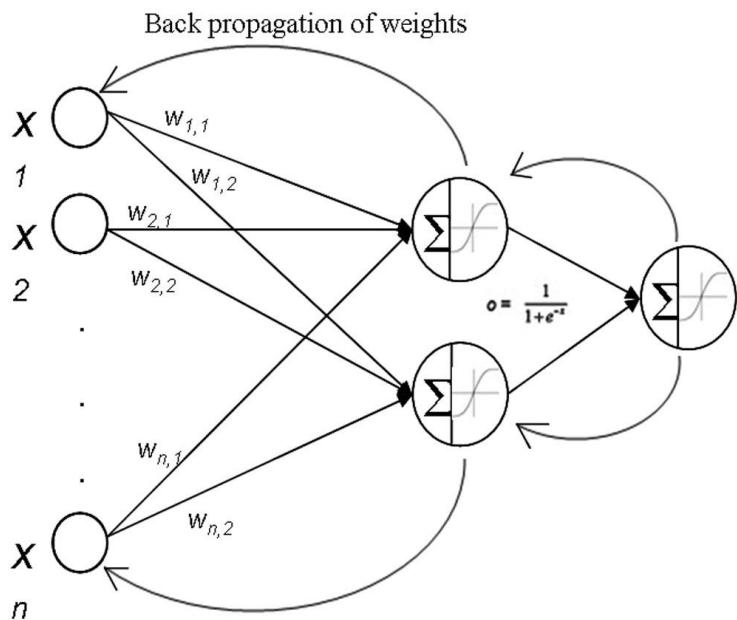
	Análise 1	Análise 2	Análise 3	Análise 4	Análise 5
Épocas	11694	11030	11494	10407	11330
EQM	0.028189	0.028252	0.028124	0.028105	0.028209
		Pesos iniciais			
w_0	0.50253	0.91858	0.68143	-0.30003	-0.29668
w_1	-0.48981	0.094431	-0.49144	-0.60681	0.66166
w_2	0.011914	-0.72275	0.62857	-0.49783	0.17053
w_3	0.39815	-0.70141	-0.51295	0.23209	0.099447
w_4	0.78181	-0.48498	0.85853	-0.05342	0.83439
		Pesos finais			
w'_0	-2.6157	-2.6152	-2.6164	-2.6166	-2.6156
w'_1	0.1715	0.17593	0.16758	0.16674	0.17307
w'_2	-1.7932	-1.7965	-1.7905	-1.79	-1.7944
w'_3	-1.6889	-1.6879	-1.6899	-1.6902	-1.6886
w'_4	-1.9486	-1.9514	-1.9463	-1.9458	-1.9496



5. MODELO : Perceptron Multicamadas (MLP)

5.1 Histórico

- Fundamentou-se no desenvolvimento do algoritmo de treinamento backpropagation.
- Foi formulado por Rumelhart, Hinton e Williams em 1986, RUMELHART et. al. (1986), precedido por propostas semelhantes ocorridas nos anos 70 e 80, WERBOS (1974); PARKER (1985)
- Mostrou que é possível treinar eficientemente Redes Neurais com camadas intermediárias



Minsky

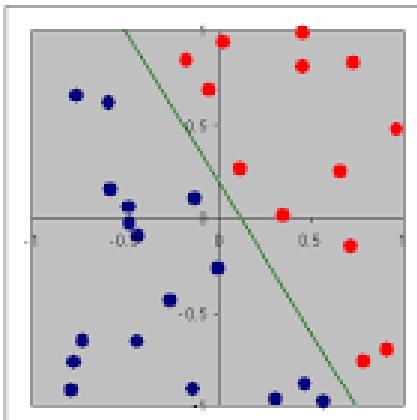


Papert

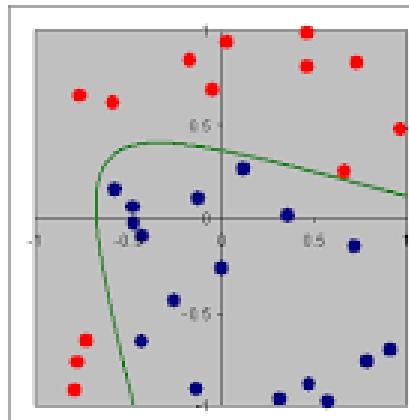


MINSKY & PAPERT (1969) demonstraram que redes de uma camada não são capazes de solucionar problemas que não sejam linearmente separáveis.

Como não acreditavam na possibilidade de se construir um método de treinamento para redes com mais de uma camada, eles concluíram que as Redes Neurais seriam sempre suscetíveis a essa limitação.



Linearmente Separável.



Não linearmente separável.



5.2. Classificadores Lineares e Não-Lineares

- EXISTEM 16 FUNÇÕES DE 2 VARIÁVEIS BOOLEANAS

		Dois padrões binários															
x	y	f ₀	f ₁	f ₂	f ₃	f ₄	f ₅	f ₆	f ₇	f ₈	f ₉	f ₁₀	f ₁₁	f ₁₂	f ₁₃	f ₁₄	f ₁₅
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1



		Um padrão binário			
x		f ₀	f ₁	f ₂	f ₃
0		0	1	0	1
1		0	0	1	1

Funções já conhecidas

$$f_0(x, y) = 0$$

$$f_3(x, y) = \bar{x} \quad (\text{não } x)$$

$$f_5(x, y) = \bar{y} \quad (\text{não } y)$$

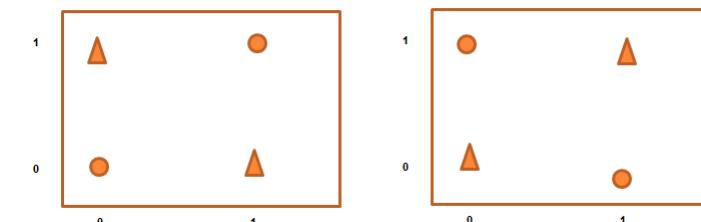
$$f_8(x, y) = x \cdot y \quad \text{AND}$$

$$f_{10}(x, y) = y$$

$$f_{12}(x, y) = x$$

$$f_{14}(x, y) = x + y \quad \text{OR}$$

$$f_{15}(x, y) = 1$$



n	nº de padrões binários	nº de funções lógicas	linearmente separáveis	% linearmente separável
1	2	4	4	100
2	4	16	14	87,5
3	8	256	104	40,6
4	16	65536	1.772	2,9
5	32	$4,3 \times 10^9$	94.572	$2,2 \times 10^{-3}$
6	64	$1,8 \times 10^{19}$	5.028.134	$3,1 \times 10^{-13}$

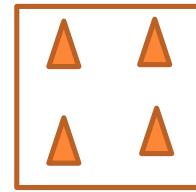
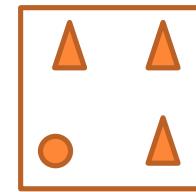
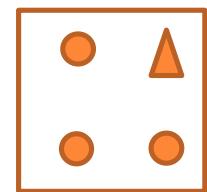
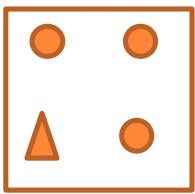
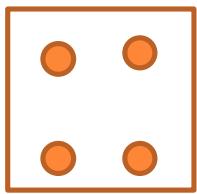
As funções lógicas de uma variável:
 $A, \bar{A}, 0, 1$

As funções lógicas de duas variáveis:
 $A, B, \bar{A}, \bar{B}, 0, 1$

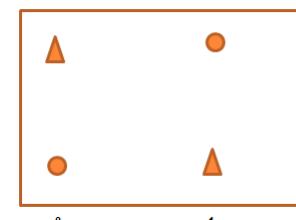
$A \vee B, A \wedge B, \bar{A} \vee B, \bar{A} \wedge B,$
 $A \vee \bar{B}, A \wedge \bar{B}, \bar{A} \vee \bar{B}, \bar{A} \wedge \bar{B}, A \oplus B, \bar{A} \oplus B$



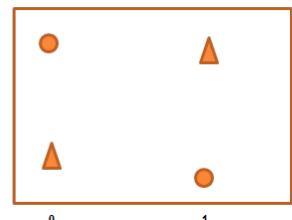
x	y	f ₀	f ₁	f ₂	f ₃	f ₄	f ₅	f ₆	f ₇	f ₈	f ₉	f ₁₀	f ₁₁	f ₁₂	f ₁₃	f ₁₄	f ₁₅
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1



f₆



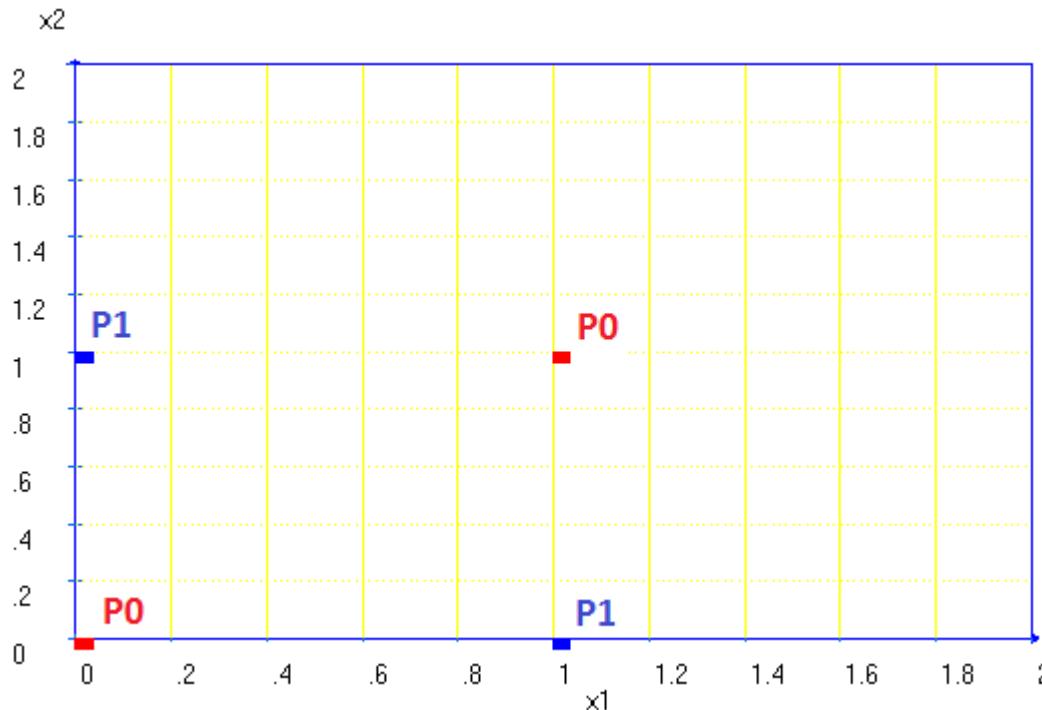
f₉



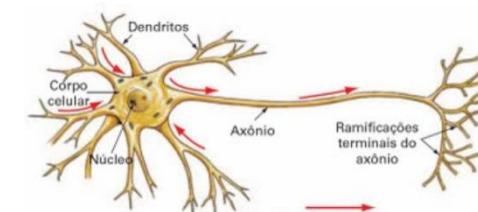
5.3. Solução para um problema não linearmente separável

Problema:

X_1	0	0	1	1
X_2	0	1	0	1
y_d	0	1	1	0

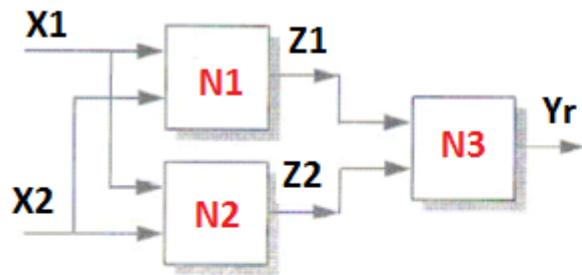


Solução ?????



x_1	0	0	1	1
x_2	0	1	0	1
y_d	0	1	1	0

Solução: Um rede com pelo menos uma camada intermediária:



$$N1 \rightarrow X_1 \cdot \overline{X}_2$$

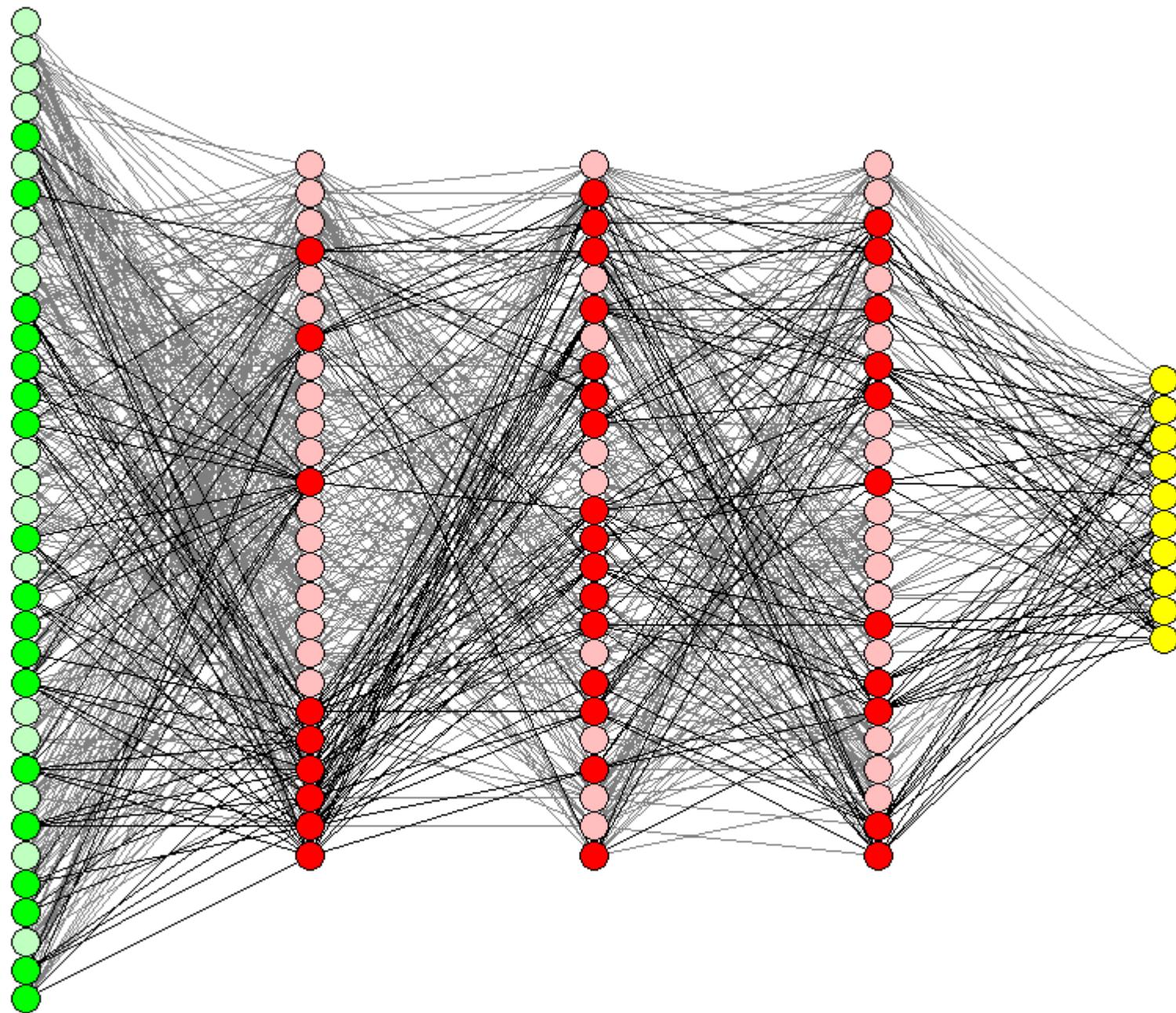
$$N2 \rightarrow \overline{X}_1 \cdot X_2$$

$$N3 \rightarrow Z1 + Z2$$

X_1	X_2	$Z1$	$Z2$	Yr	y_d
0	0	0	0	0	0
0	1	0	1	1	1
1	0	1	0	1	1
1	1	0	0	0	0

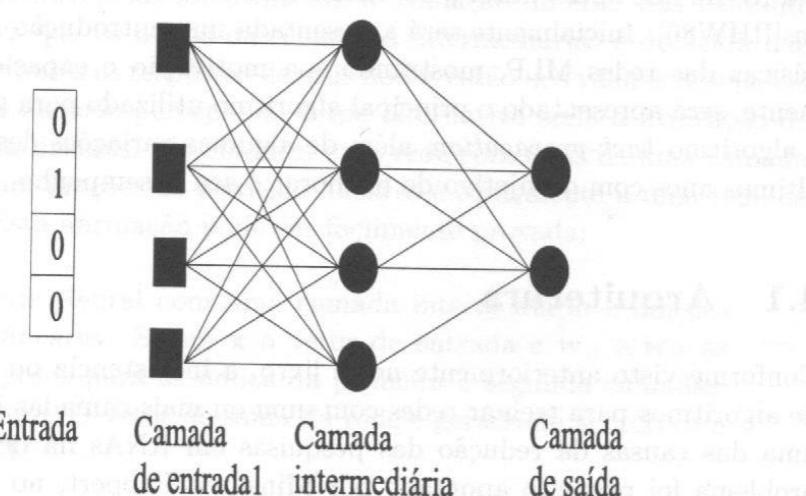
Problema adicional: Como ajustar pesos intermediário de forma automática:





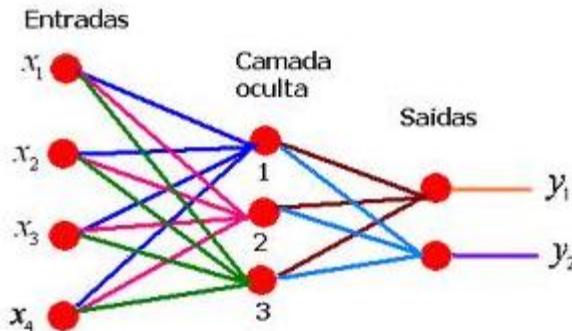
5.4. Estrutura de rede do PMC

- Cada camada recebe dados da camada imediatamente inferior e envia para a camada subsequente.
- Não existem conexões entre elementos da mesma camada.
- Cada camada tem uma função específica.
- A camada de saída recebe os estímulos das camadas intermediárias e constrói o padrão que será a resposta.
- As camadas intermediárias funcionam como extratoras de características, seus pesos são uma codificação de características apresentadas nos padrões de entrada e permitem que a rede crie sua própria representação, mais rica e complexa, do problema.



5.5. Número de camadas intermediárias (ocultas)

- Apenas uma camada intermediária é suficiente para aproximar qualquer função contínua - CYBENKO (1989) .
- Para a resolução de **problemas de classificação** uma rede neural com uma camada escondida é mais que suficiente.
- CYBENKO (1988) provou, a partir de extensões do Teorema de Kolmogoroff, que são necessárias no máximo duas camadas intermediárias, com um número suficiente de unidades (neurônios) por camada, para se produzirem quaisquer **mapeamentos (ajuste)**.
- Para problemas muito complexos a utilização de três camadas intermediárias pode facilitar o treinamento ou melhorar a capacidade de generalização das redes MLP.



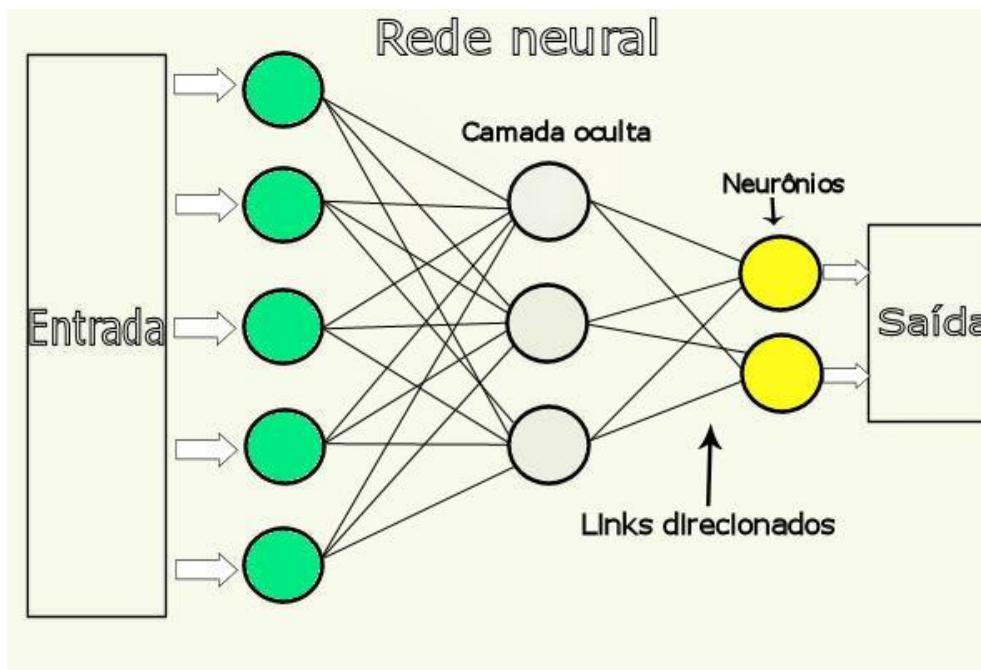
5.6 Número de neurônios por camada

Cuidados preliminares:

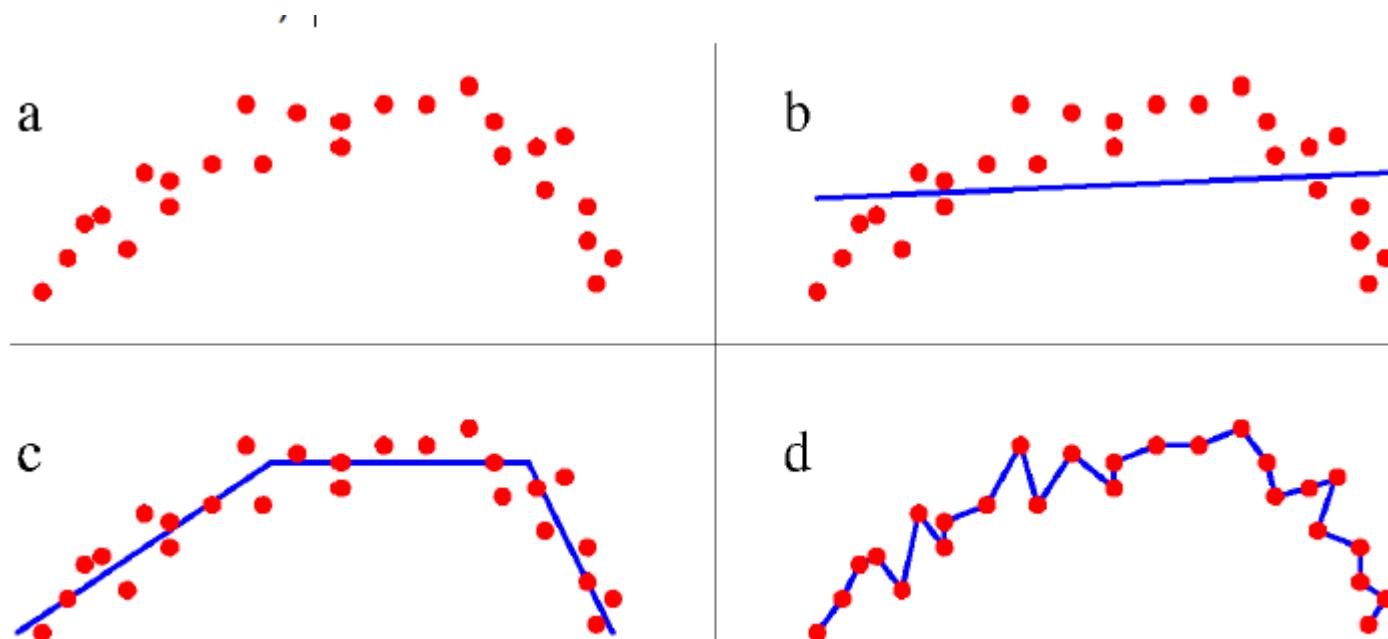
É geralmente definido EMPIRICAMENTE.

Deve-se ter cuidado para:

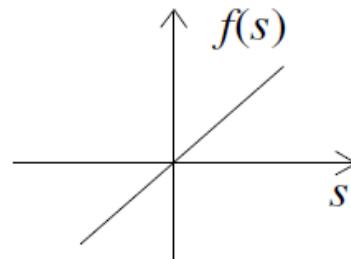
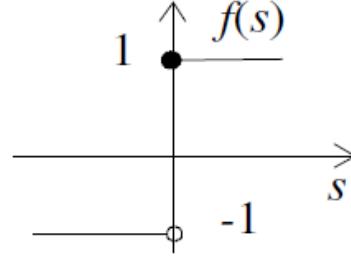
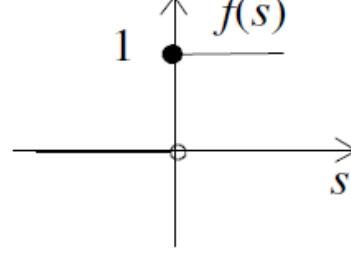
- não utilizar nem unidades demais, o que pode levar a rede a memorizar os dados de treinamento (overfitting), ao invés de extrair as características gerais que permitirão a generalização
- Não utilizar um número muito pequeno, que pode forçar a rede a gastar tempo em excesso tentando encontrar uma representação ótima.



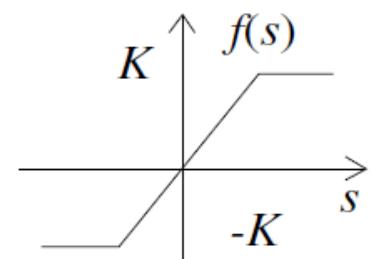
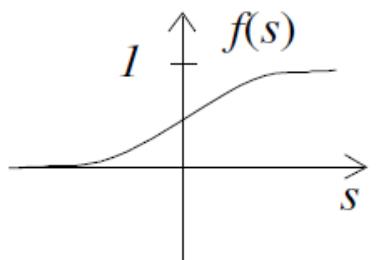
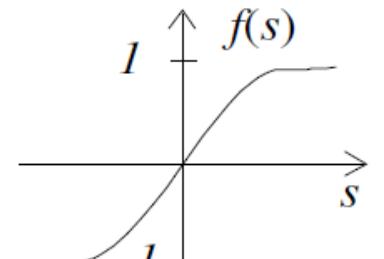
- Overfitting: acontece quando um no. excessivo de neurônios é usado!
 - Depois de um certo ponto do treinamento, a rede piora ao invés de melhorar.
 - Rede “memoriza” padrões de treinamento, incluindo todas as suas peculiaridades (ruído).
- Exemplos de soluções:
 - Encerrar treinamento cedo (*early-stopping*).
 - Adoção das técnicas de *pruning* (*eliminação de pesos e nodos irrelevantes*).



5.7. Uso de funções de ativação

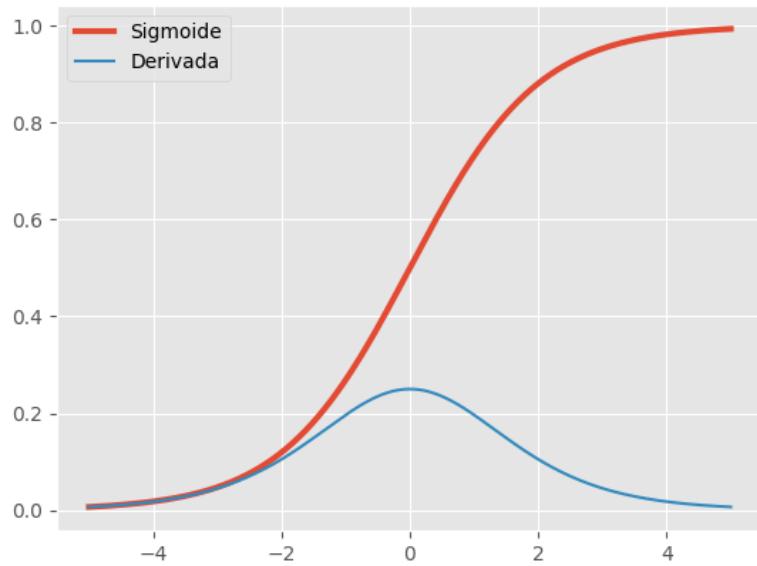
Linear	$f(s) = s$	purelin	
Sinal	$f(s) = \begin{cases} +1 & \text{se } s \geq 0 \\ -1 & \text{se } s < 0 \end{cases}$	hardlims	
Degrau	$f(s) = \begin{cases} +1 & \text{se } s \geq 0 \\ 0 & \text{se } s < 0 \end{cases}$	hardlim	



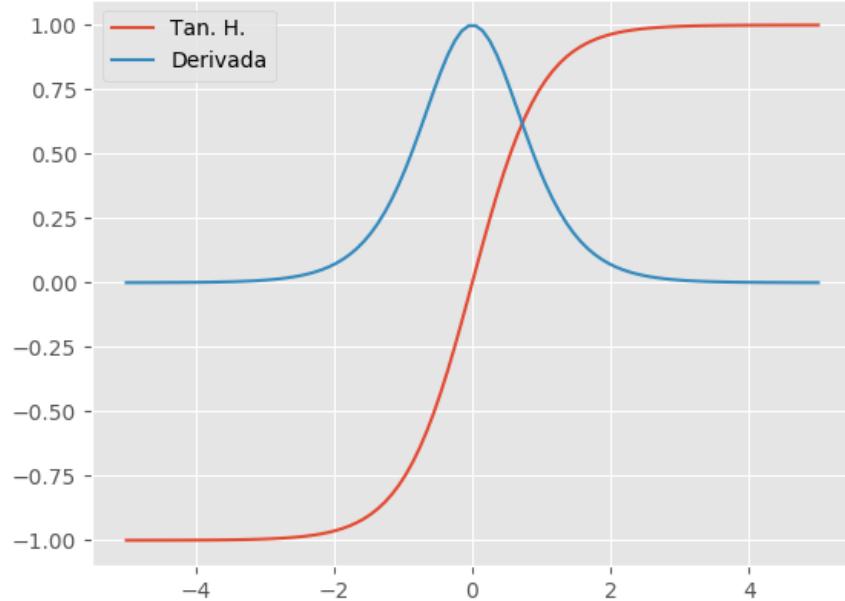
BSB Limiar Lógico	ou $f(s) = \begin{cases} -K & \text{se } s \leq -K \\ s & \text{se } -K < s < +K \\ +K & \text{se } s \geq +K \end{cases}$	satlin satlins	
Logística	$f(s) = \frac{1}{1 + e^{-s}}$	logsig	
Tangente Hiperbólica	$f(s) = \tanh(s) = \frac{1 - e^{-2s}}{1 + e^{-2s}}$	tansig	



$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad \sigma'(x) = \sigma(x)(1 - \sigma(x))$$



$$\tanh(x) = 2\sigma(2x) - 1 \quad \tanh'(x) = 1 - \tanh^2(x)$$



5.8. Treinamento

Algoritmo *Error Backpropagation*

- Treinamento é feito em duas fases:

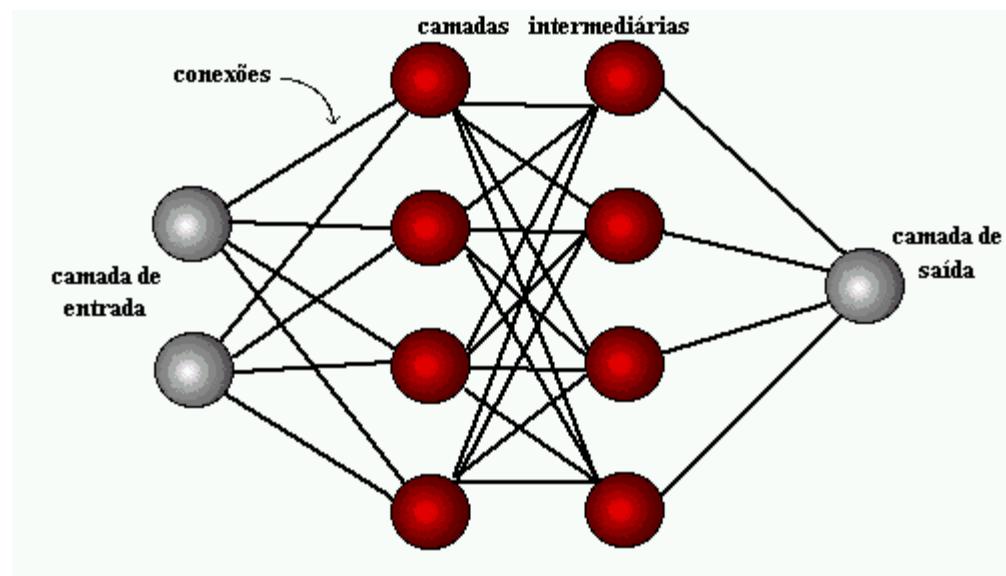
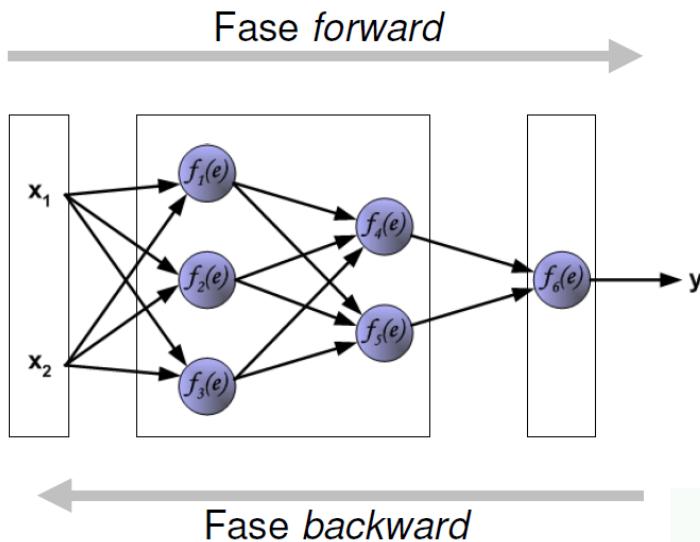
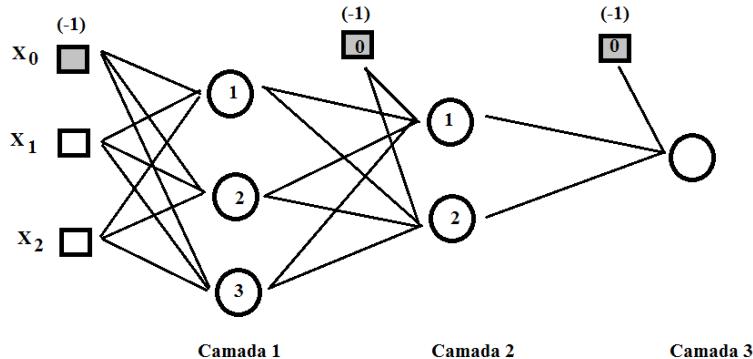


Figura 5. Arquitetura de uma rede Perceptron multicamadas, com duas camadas intermediárias.



Camada	1	2	3 (saída)
Núm. de neurônios	n_1	n_2	n_3
Núm. de entradas	n_0+1	n_1+1	n_2+1

n_0 : número de entradas, no exemplo é igual a 2;

n_1 : número de neurônios na camada oculta 1, no exemplo igual a 3;

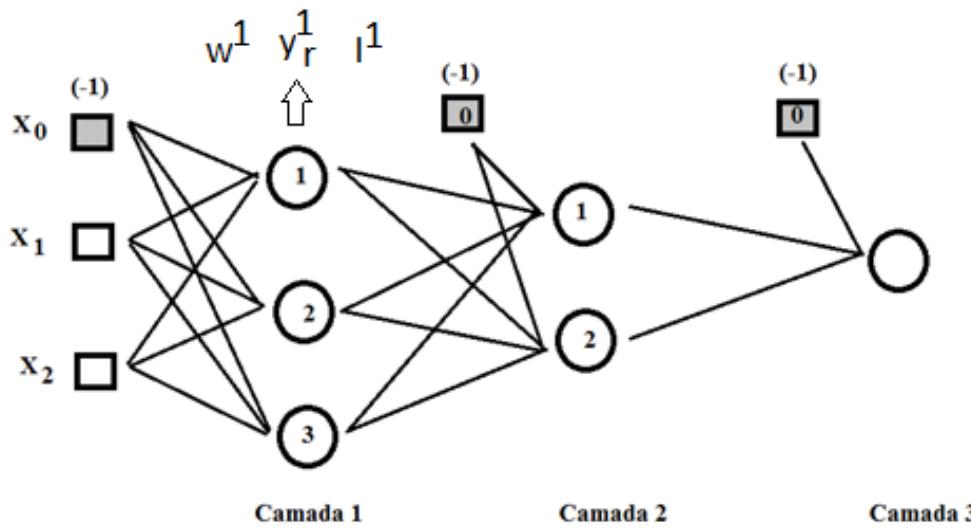
n_2 : número de neurônios na camada oculta 2, no exemplo igual a 2;

n_3 : número de neurônios na camada de saída, no exemplo igual a 1;



Fase forward

Passo 1. Nesta fase as entradas são apresentadas à primeira camada da rede e propagadas em direção às saídas.



A seguinte matriz de pesos sinápticos a ser estimados para esta camada pode ser definida por:

$$w_{N,E}^1 = \begin{bmatrix} w_{10}^1 & w_{11}^1 & w_{12}^1 \\ w_{20}^1 & w_{21}^1 & w_{22}^1 \\ w_{30}^1 & w_{31}^1 & w_{32}^1 \end{bmatrix}_{n1x(n0+1)=3x3}$$

Os potenciais de ativação, I , e as saídas de rede, y_r , podem ser definidas por meio de:

$$I^1 = w_{N,E}^1 x = \begin{bmatrix} I_1^1 \\ I_2^1 \\ I_3^1 \end{bmatrix}_{n1x1=3x1} \quad \text{e} \quad y_r^1 = \begin{bmatrix} -1 \\ \dots \\ g(I^1) \end{bmatrix}_{(n1+1)x1=4x1}$$

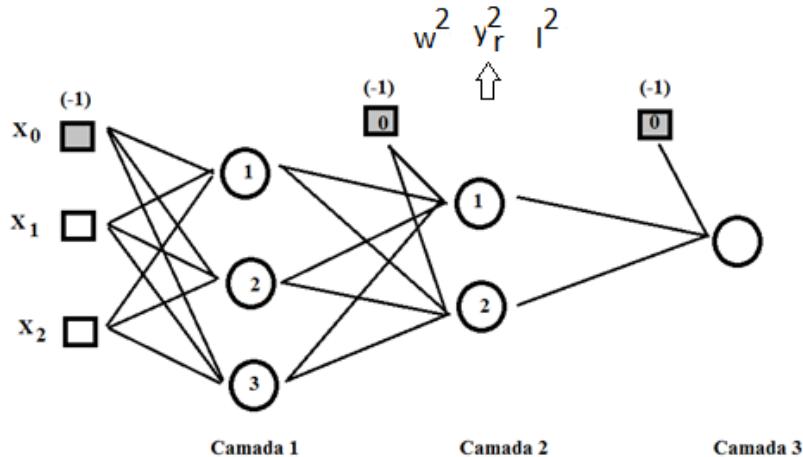
Sendo x o vetor de entrada dado por:

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}_{(n0+1)x1=3x1}$$

E $g(I^1)$ o valor obtido pela aplicação da função de ativação



Passo 2: Neste etapa são calculados, para cada neurônio da camada i (camada 1), os sinais de saída e propagam para a camada $i+1$ (camada 2).



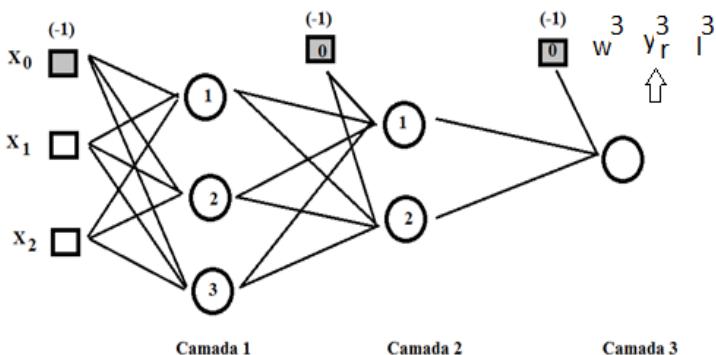
A seguinte matriz de pesos sinápticos a ser estimadas para esta camada pode ser definida por:

$$w_{N,E}^2 = \begin{bmatrix} w_{10}^2 & w_{11}^2 & w_{12}^2 & w_{13}^2 \\ w_{20}^2 & w_{21}^2 & w_{22}^2 & w_{23}^2 \end{bmatrix}_{n2 \times (n1+1)=2 \times 4}$$

Os potenciais de ativação, l , e as saídas de rede, y_r , podem ser definidos por meio de:

$$I^2 = w_{N,E}^2 y_r^1 = \begin{bmatrix} I_1^2 \\ I_2^2 \end{bmatrix}_{n2 \times 1=2 \times 1} \quad \text{e} \quad y_r^2 = \begin{bmatrix} -1 \\ \dots \\ g(I^2) \end{bmatrix}_{(n2+1) \times 1=3 \times 1}$$

Passo 3: Nesta etapa, atinge-se a última camada (camada de saída) em que são calculadas as respectivas saídas cujos valores serão comparados à saída desejada.



A seguinte matriz de pesos sinápticos a ser estimadas para esta camada pode ser definida por:

$$w_{N,E}^3 = [w_{10}^3 \quad w_{11}^3 \quad w_{12}^3]_{n_3 \times (n_2 + 1) = 1 \times 3}$$

Os potenciais de ativação, I , e as saídas de rede, y_r , podem ser definidas por meio de:

$$I^3 = w_{N,E}^3 y_r^2 = [I_1^3]_{n_3 \times 1 = 1 \times 1} \quad \text{e} \quad y_r^3 = [g(I^3)]_{n_3 \times 1 = 1 \times 1}$$

Passo 4: Finalização da etapa forward em que se calcula, na camada de saída, o erro da rede.

O erro de saída é dado por:

$$\delta = (y - y_r)$$

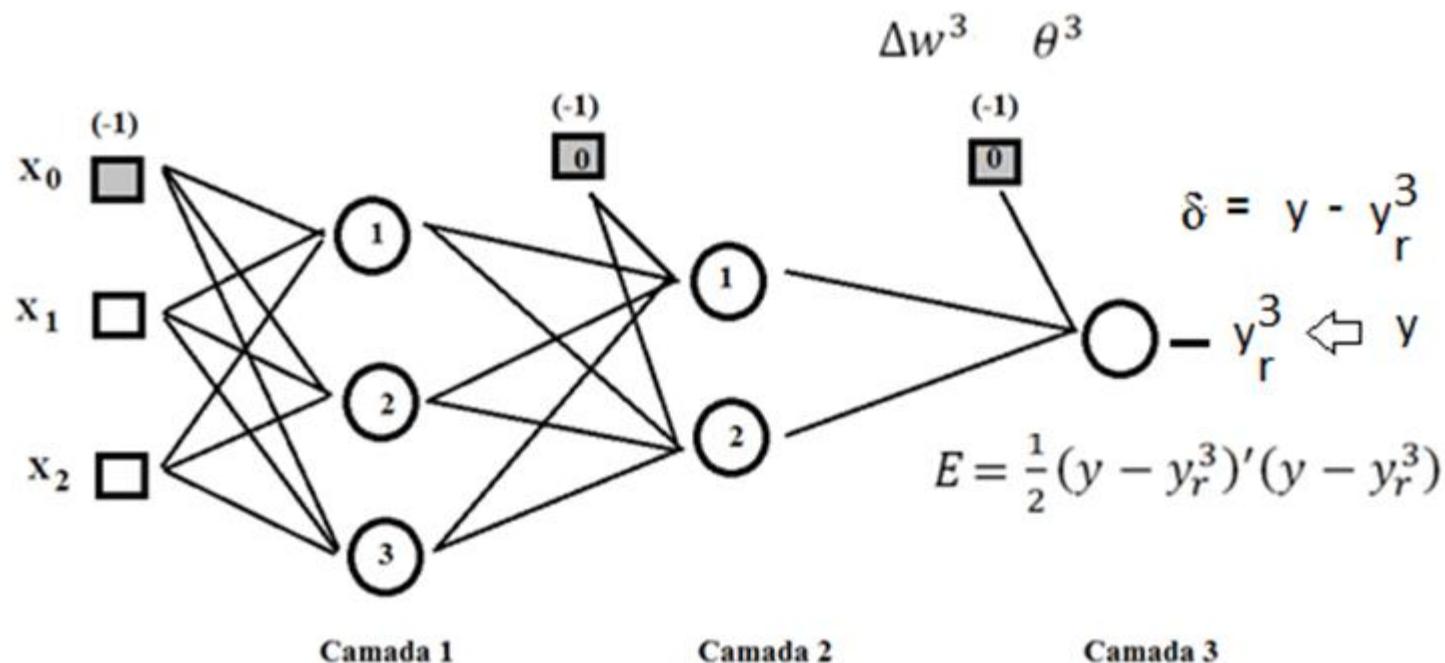
Uma função de custo a ser minimizada é estabelecida por:

$$E = (1/2) \delta' \delta = (1/2)(y - y_r)'(y - y_r)$$



Fase backward

Passo 1: Ajuste dos pesos na terceira camada – camada de saída. O valor do erro é propagado para as camadas anteriores (usando os pesos respectivos como parâmetro). O erro de cada camada é calculado



O ajuste é realizado empregando a abordagem do gradiente descendente e a regra de diferenciação em cadeia. Podemos assumir que:

1 2 3

Como $I^3 = w_{N,E}^3 y_r^2$

$$\frac{\delta E}{\delta w_{N,E}^3} = \frac{\delta E}{\delta y_r^3} \frac{\delta y_r^3}{\delta I^3} \frac{\delta I^3}{\delta w_{N,E}^3}$$

$$\frac{\delta I^3}{\delta w_{N,E}^3} = y_r^2 \quad 3$$

Como $y_r^3 = [g(I^3)]$

então $\frac{\delta y_r^3}{\delta I^3} = g'(I^3) \quad 2 \quad |$

Sabendo que:

$$E = \frac{1}{2} (y - y_r^3)' (y - y_r^3) = \frac{1}{2} \sum_{j=1}^n (y_j - y_{rj}^3)^2 \quad \text{então} \quad \frac{\delta E}{\delta y_r^3} = -(y - y_r^3) \quad 1 \quad ||$$

Concluímos que:

$$\frac{\delta E}{\delta w_{N,E}^3} = \frac{\delta E}{\delta y_r^3} \frac{\delta y_r^3}{\delta I^3} \frac{\delta I^3}{\delta w_{N,E}^3} = -(y - y_r^3) g'(I^3) y_r^2 \quad 1 \quad 2 \quad 3$$

Logo, o ajuste da matriz de pesos da terceira camada, denotado por $w_{N,E}^3$, ajustado em direção oposta ao gradiente para os erros sejam minimizados, é realizado por meio da expressão:

$$\Delta w_{N,E}^3 = -\eta \frac{\delta E}{\delta w_{N,E}^3} \quad \text{logo} \quad \Delta w_{N,E}^3 = \eta (y - y_r^3) g'(I^3) y_r^2 = \eta \theta^3 y_r^2$$

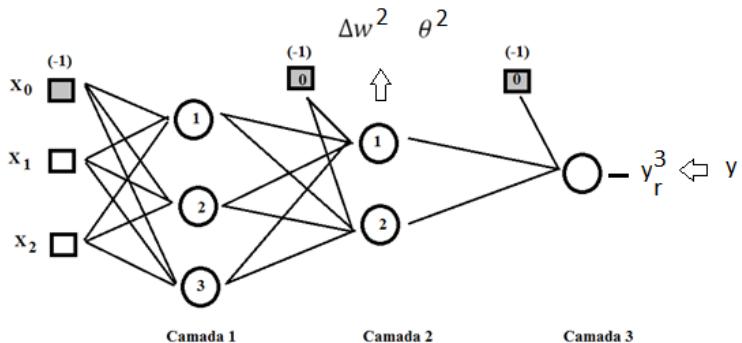
m que

$\theta^3 = (y - y_r^3) g'(I^3)$ é o gradiente local em relação ao neurônio da camada de saída.

Assim, a regra de ajuste é:

$$w_{N,E(t+1)}^3 = w_{N,E(t)}^3 + \eta \theta^3 y_r^2$$

Passo 2: Ajuste dos pesos na segunda camada – camada oculta ou intermediária



O ajuste é novamente realizado empregando a abordagem do gradiente descendente e a regra de diferenciação em cadeia. Podemos assumir que:

$$\frac{\delta E}{\delta w_{N,E}^2} = \frac{1}{\delta y_r^2} \frac{2}{\delta I^2} \frac{3}{\delta w_{N,E}^2}$$

Como $I^2 = w_{N,E}^2 y_r^1$

então $\frac{\delta I^2}{\delta w_{N,E}^2} = y_r^1 \quad 3$

Como $y_r^2 = [g(I^2)]$

(excluindo o elemento associado ao limiar)

então

$$\frac{\delta y_r^2}{\delta I^2} = g'(I^2) \quad 2$$

III

Verifica-se que $\frac{\delta E}{\delta y_r^2} = \frac{\delta E}{\delta I^3} \frac{\delta I^3}{\delta y_r^2} = \frac{\delta E}{\delta I^3} \frac{\delta (w_{N,E}^3 y_r^2)}{\delta y_r^2} = \frac{\delta E}{\delta I^3} w_{N,E}^3 \quad IV$



Já vimos que:

$$\frac{\delta y_r^3}{\delta I^3} = g'(I^3) \quad | \quad \text{e} \quad \frac{\delta E}{\delta y_r^3} = -(y - y_r^3) \quad ||$$

Portanto, o produto destas duas expressões resulta em:

$$\frac{\delta y_r^3}{\delta I^3} \frac{\delta E}{\delta y_r^3} = g'(I^3) [-(y - y_r^3)]$$

logo

$$\frac{\delta E}{\delta I^3} = -(y - y_r^3)g'(I^3)$$

e

$$\frac{\delta E}{\delta y_r^2} = \frac{\delta E}{\delta I^3} w_{N,E}^3 = -(y - y_r^3)g'(I^3)w_{N,E}^3 \quad 1$$

Concluímos que:

$$\frac{\delta E}{\delta w_{N,E}^2} = \frac{\delta E}{\delta y_r^2} \frac{\delta y_r^2}{\delta I^2} \frac{\delta I^2}{\delta w_{N,E}^2} = -\underbrace{(y - y_r^3)g'(I^3)w_{N,E}^3}_{1} \underbrace{g'(I^2)}_{2} \underbrace{y_r^1}_{3}$$

Logo, o ajuste da matriz de pesos da segunda camada, denotado por $w_{N,E}^2$, ajustado em direção oposta ao gradiente para os erros sejam minimizados, é realizado por meio da expressão:

$$\Delta w_{N,E}^2 = -\eta \frac{\delta E}{\delta w_{N,E}^2} \quad \text{logo} \quad \Delta w_{N,E}^2 = \eta(y - y_r^3)g'(I^3)w_{N,E}^3g'(I^2)y_r^1 = \eta\theta^2y_r^1$$

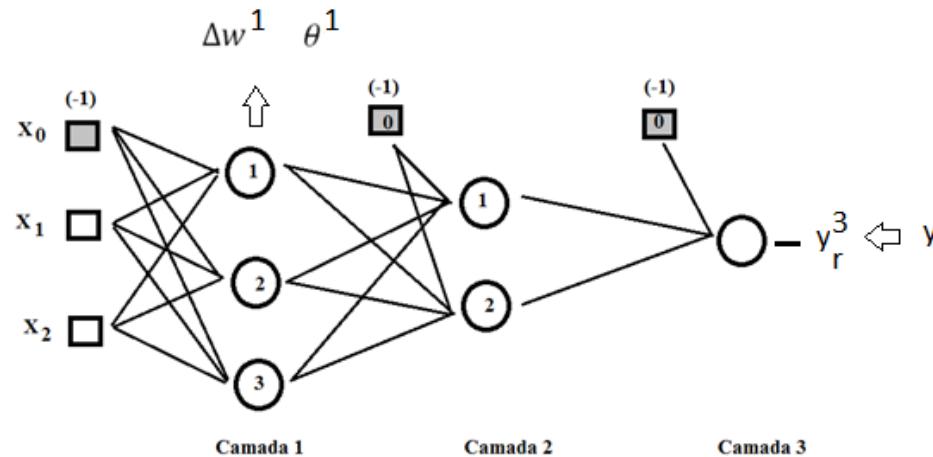
em que

$\theta^2 = (y - y_r^3)g'(I^3)w_{N,E}^3g'(I^2) = \theta^3 w_{N,E}^3 g'(I^2)$ é o gradiente local em relação ao neurônio da camada de intermediária.

Assim, a regra de ajuste é:

$$w_{N,E(t+1)}^2 = w_{N,E(t)}^2 + \eta\theta^2y_r^1$$

Passo 3: Ajuste dos pesos na primeira camada – camada oculta ou intermediária



O ajuste é novamente realizado empregando a abordagem do gradiente descendente e a regra de diferenciação em cadeia. Podemos assumir que:

$$\frac{\delta E}{\delta w_{N,E}^1} = \frac{1}{\delta y_r^1} \frac{2}{\delta I^1} \frac{3}{\delta w_{N,E}^1}$$

Como $I^1 = w_{N,E}^1 x$ então $\frac{\delta I^1}{\delta w_{N,E}^1} = x$ 3

Como $y_r^1 = [g(I^1)]$ (excluindo o elemento associado ao limiar) então

$$\frac{\delta y_r^1}{\delta I^1} = g'(I^1) \quad \text{2}$$

Verifica-se que $\frac{\delta E}{\delta y_r^1} = \frac{\delta E}{\delta I^2} \frac{\delta I^2}{\delta y_r^1} = \frac{\delta E}{\delta I^2} \frac{\delta(w_{N,E}^2 y_r^1)}{\delta y_r^1} = \frac{\delta E}{\delta I^2} w_{N,E}^2$

Já vimos que:

$$\frac{\delta y_r^2}{\delta I^2} = g'(I^2) \quad \text{III} \quad \text{e} \quad \frac{\delta E}{\delta y_r^2} = \frac{\delta E}{\delta I^3} w_{N,E}^3 = -(y - y_r^3) g'(I^3) w_{N,E}^3 \quad \text{IV}$$

Portanto, o produto destas duas expressões resulta em:

$$\frac{\delta y_r^2}{\delta I^2} \frac{\delta E}{\delta y_r^2} = g'(I^2) [-(y - y_r^3) g'(I^3) w_{N,E}^3]$$

logo

$$\frac{\delta E}{\delta I^2} = -(y - y_r^3) g'(I^3) w_{N,E}^3 g'(I^2)$$

e

$$\frac{\delta E}{\delta y_r^1} = \frac{\delta E}{\delta I^2} w_{N,E}^2 = -(y - y_r^3) g'(I^3) w_{N,E}^3 g'(I^2) w_{N,E}^2 \quad \text{1}$$

Concluímos que:



$$\frac{\delta E}{\delta w_{N,E}^1} = \frac{\delta E}{\delta y_r^1} \frac{\delta y_r^1}{\delta I^1} \frac{\delta I^1}{\delta w_{N,E}^1} = - \underbrace{(y - y_r^3) g'(I^3)}_1 w_{N,E}^3 g'(I^2) w_{N,E}^2 g'(I^1) x$$

2 3

Logo, o ajuste da matriz de pesos da segunda camada, denotado por $w_{N,E}^1$, ajustado em direção oposta ao gradiente para os erros sejam minimizados, é realizado por meio da expressão:

$$\Delta w_{N,E}^1 = -\eta \frac{\delta E}{\delta w_{N,E}^1} \quad \text{logo}$$

$$\Delta w_{N,E}^1 = \eta (y - y_r^3) g'(I^3) w_{N,E}^3 g'(I^2) w_{N,E}^2 g'(I^1) x = \eta \theta^1 x$$

Lembrando que

$$\theta^3 = (y - y_r^3) g'(I^3)$$

$$\theta^2 = (y - y_r^3) g'(I^3) w_{N,E}^3 g'(I^2)$$

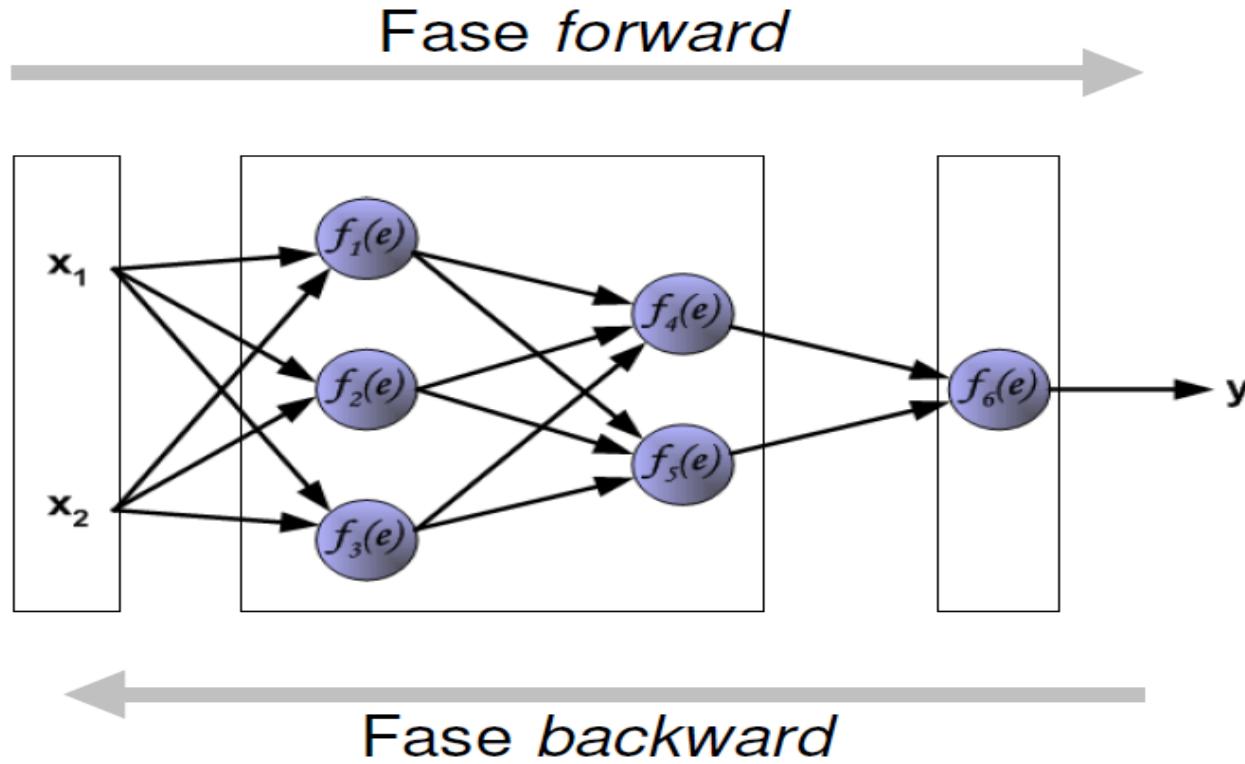
e

$$\theta^1 = (y - y_r^3) g'(I^3) w_{N,E}^3 g'(I^2) w_{N,E}^2 g'(I^1) = \theta^2 w_{N,E}^2 g'(I^1)$$

é o gradiente local em relação ao neurônio da primeira camada.
Assim, a regra de ajuste é:

$$w_{N,E(t+1)}^1 = w_{N,E(t)}^1 + \eta \theta^1 x$$

Repete-se o processo enquanto enquanto a rede não aprender o mapeamento entrada-saída desejada.



Desvantagens do algoritmo de aprendizagem *backpropagation*:

- Normalmente o tempo de processamento é elevado
- A arquitetura da rede deve ser fixada *a priori*



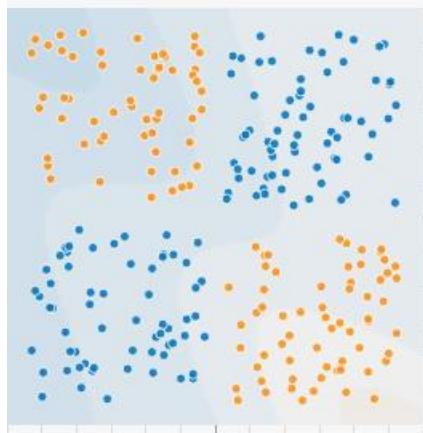
5.9 Considerações finais: Fases do projeto de MLPs

- Estudo do problema
 - Construção de um conjunto de treinamento
 - Conjunto de dados para validação
 - Conjunto de dados para teste
- Escolha de uma arquitetura
 - Número de camadas ocultas
 - Número de neurônios por camada
 - Funções de ativação
- Escolha de um algoritmo de aprendizado
- Parâmetros a se ajustar
 - Número de épocas
 - Erro quadrático médio
 - Critério de otimização

A seleção dos parâmetros da rede é um processo tão pouco compreendido que é muitas vezes chamado de “**magia negra**”. Pequenas diferenças nestes parâmetros podem levar a grandes diferenças tanto no tempo de treinamento como na generalização obtida.

5.10 Atividade

<http://playground.tensorflow.org/#activation=tanh&batchSize=10&dataset=circle®Dataset=reg-plane&learningRate=0.03®ularizationRate=0&noise=0&networkShape=4,2&seed=0.44234&showTestData=false&discretize=false&percTrainData=50&x=true&y=true&xTimesY=false&xSquared=false&ySquared=false&cosX=false&sinX=false&cosY=false&sinY=false&collectStats=false&problem=classification&initZero=false&hideText=false>



Quantas camadas e neurônios/camada seriam necessários para discriminar estas populações?

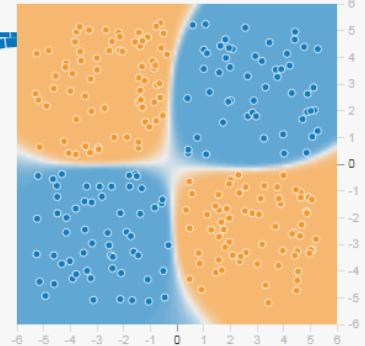
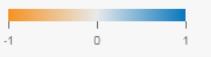
DATA
Which dataset do you want to use?

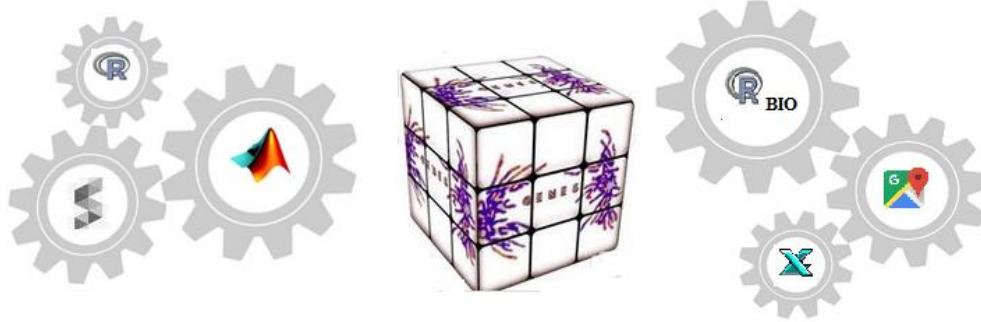



Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE

FEATURES
Which properties do you want to feed in?
 X_1 X_2 X_1^2 X_2^2 $X_1 X_2$ $\sin(X_1)$ $\sin(X_2)$

HIDDEN LAYERS
+ - 2 HIDDEN LAYERS
+ - 4 neurons
+ - 2 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.

OUTPUT
Test loss 0.003
Training loss 0.000

Colors shows data, neuron and weight values. 



PARTE II - APLICAÇÕES

 **II WORKSHOP EM INTELIGÊNCIA COMPUTACIONAL E
APRENDIZADO ESTATÍSTICO APLICADOS À AGROPECUÁRIA**
10 A 13 DE SETEMBRO

MINICURSOS

MINICURSO 1: REDES NEURAIS UTILIZANDO O SOFTWARE GENES
MINISTRANTE: PROF. COSME DAMIÃO CRUZ -DEPARTAMENTO DE BIOLOGIA
GERAL /UFV
DATA: 10 DE SETEMBRO DE 2019
HORÁRIO: 8 :00 - 12:00
LOCAL: LABORATÓRIO DE BIOINFORMÁTICA- BIOAGRO/UFV- SALA 109



PROBLEMAS QUEM PODEM SER RESOLVIDOS POR MEIO DA IC

- Classificação
- Ajuste de Modelos e Predição



APLICAÇÃO 1 – ANÁLISE DE CLASSIFICAÇÃO

Problema: Foram avaliadas 6 populações em relação a 9 características.

Cada população foi representada por 400 indivíduos.

Objetivo: estudar a diferenciação entre as populações e estabelecer critério que permita classificar novos indivíduos em uma das seis populações consideradas.

Arquivos:

dados6p.dat

dados6p_t.dat

dados6p_v.dat

dados6pRP.dat

1	2609.7861	1419.3375	1940.8091	2579.8815	1755.6009	2216.5698	5700.0062	5761.9911	5421.5396
1	3265.1171	2411.3743	1845.9314	2620.394	1719.1102	1778.0594	3712.7261	5941.009	6359.7513
1	1903.2267	2009.1737	1903.3327	2373.2394	1669.402	1885.8464	7203.5577	6881.0499	5579.275
1	720.3046	2185.7629	2156.929	2999.7513	2342.4266	1686.1869	6602.3291	7472.8863	6048.3604
1	1942.1534	2241.9356	1891.4429	2348.3518	1776.8075	2482.2731	6850.2942	8351.1201	5697.7585
1	2997.2632	1498.04	2374.4183	1948.0134	1416.4597	1940.0746	4454.7952	5881.7313	6494.8133
1	2144.9418	2160.1315	2090.5845	2494.4419	1897.9777	2094.4905	8786.1794	6262.855	6242.169
1	1961.5416	2840.2902	2244.5233	1121.2262	2886.4181	1759.14	8845.6022	6591.108	5331.6336
1	1614.0394	1343.5398	2082.9907	976.8351	1221.8049	1876.4112	5760.6163	4857.3545	5446.0814

Estudo 1. Reconhecimento de Padrões

a. Componentes Principais

A análise por componentes principais consiste em transformar um conjunto original de variáveis em outro conjunto de dimensão equivalente, mas com propriedades importantes, que são de grande interesse em certos estudos de melhoramento.

Cada componente principal é uma combinação linear das variáveis originais. Além disso, são independentes entre si e estimados com o propósito de reter, em ordem de estimação, o máximo da informação, em termos de variação total, contida nos dados iniciais.

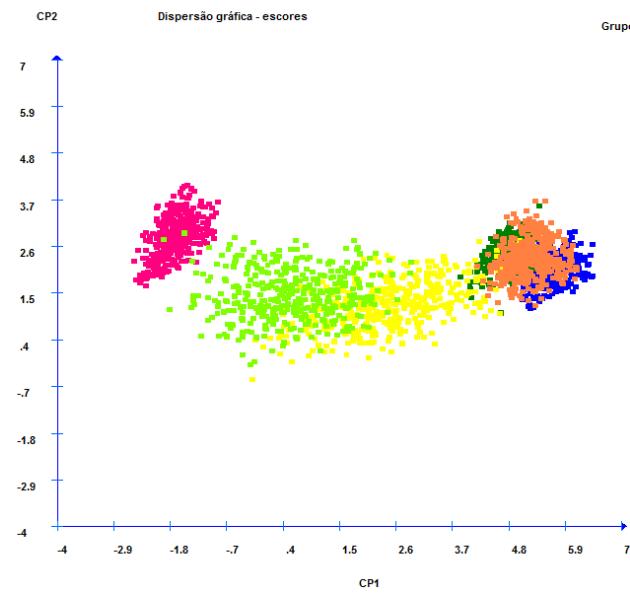
$$CP_1 = a_{11}x_1 + a_{21}x_2 + \dots + a_{v1}x_v$$

...

$$CP_n = a_{1n}x_1 + a_{2n}x_2 + \dots + a_{vn}x_v$$

$$(R - I\lambda_k)\alpha_k = \phi$$

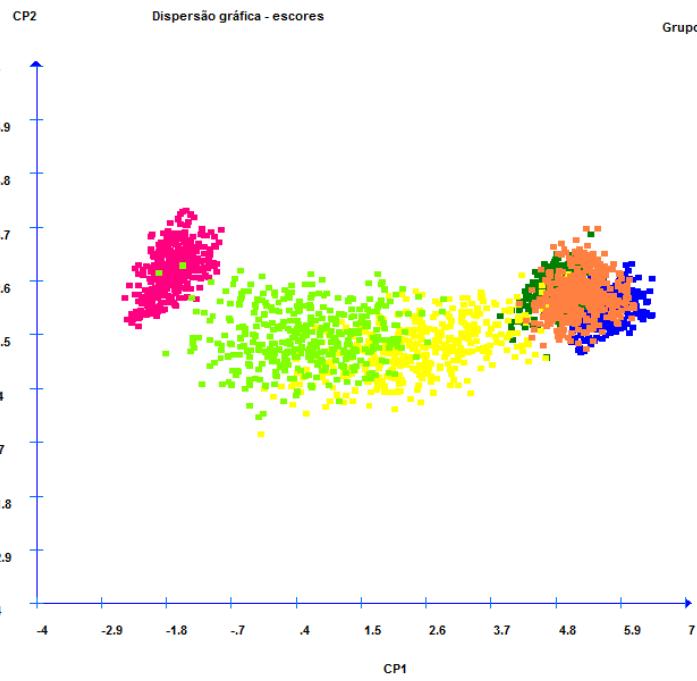
$$\left. \begin{array}{l} V(CP_1) = \lambda_1 \\ \vdots \\ V(CP_n) = \lambda_n \end{array} \right\}$$



Estudo 1. Reconhecimento de Padrões

MATRIZ DE CORRELAÇÃO								
1	0.8663	0.8861	0.7745	0.8475	0.8682	0.6986	0.7264	0.761
0.8663	1	0.9477	0.8619	0.9246	0.9441	0.7331	0.7653	0.8197
0.8861	0.9477	1	0.8651	0.9378	0.9496	0.7503	0.787	0.83
0.7745	0.8619	0.8651	1	0.8632	0.8994	0.7172	0.7324	0.7896
0.8475	0.9246	0.9378	0.8632	1	0.9474	0.7246	0.7482	0.8056
0.8682	0.9441	0.9496	0.8994	0.9474	1	0.7613	0.7544	0.8322
0.6986	0.7331	0.7503	0.7172	0.7246	0.7613	1	0.7541	0.8191
0.7264	0.7653	0.787	0.7324	0.7482	0.7544	0.7541	1	0.837
0.761	0.8197	0.83	0.7896	0.8056	0.8322	0.8191	0.837	1

RAIZ	RAIZ (%)	Acum.
$V(CP_1) = \lambda_1$	7.57863584.2070684.20706	
	0.5036245.59581789.80288	
	0.2514632.79403592.59691	
	0.2281512.53501695.13193	
	0.1437491.59721696.72914	
	0.1327681.47520498.20435	
	0.0745460.82829299.03264	
$V(CP_n) = \lambda_n$	0.0471440.52382499.55646	
	0.0399190.443539	100

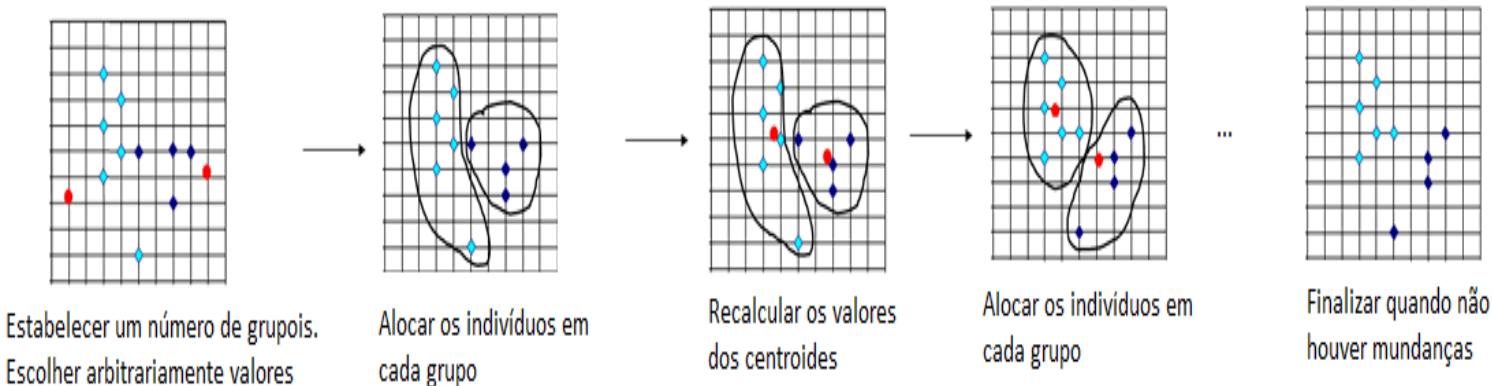


$$CP_{\eta} = a_{1\eta}x_1 + a_{2\eta}x_2 + \dots + a_{v\eta}x_v$$

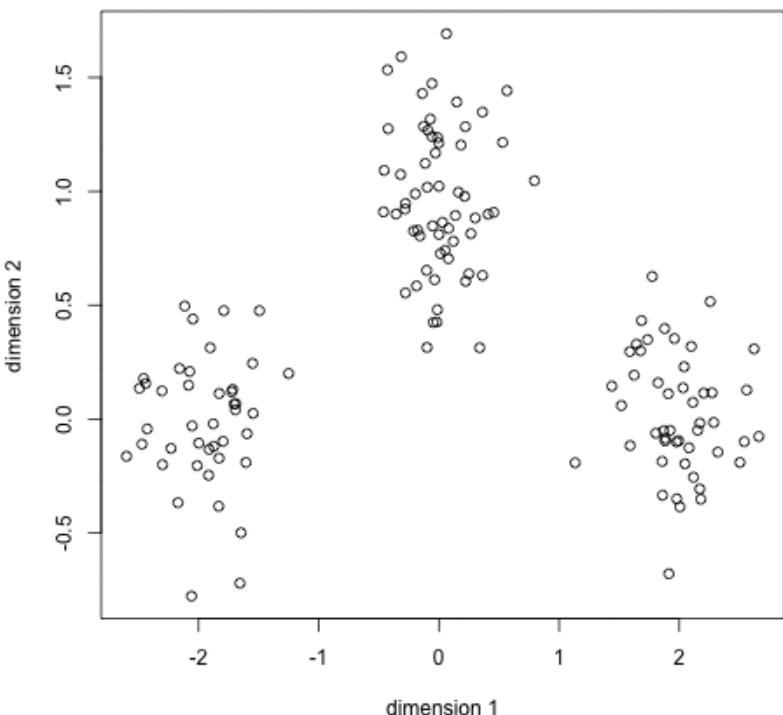
$$CP_1 = a_{11}x_1 + a_{21}x_2 + \dots + a_{v1}x_v$$

CONJUNTO	DE	AUTOVETORES	ASSOCIADOS						
0.3274	0.347	0.3509	0.3307	0.3442	0.351	0.3047	0.3115	0.3292	
-0.2257	-0.2344	-0.2022	-0.1596	-0.2653	-0.2285	0.5683	0.4847	0.3837	
-0.2438	-0.0316	-0.067	0.2559	0.0424	0.1443	0.623	-0.6758	-0.0453	
0.6768	0.0125	0.0709	-0.6086	-0.0664	-0.0646	0.3224	-0.1944	-0.1265	
-0.3295	0.1982	0.1157	-0.5235	0.2317	0.0927	-0.1892	-0.2819	0.6222	
0.4393	-0.2131	-0.2048	0.3552	-0.3426	-0.085	-0.2297	-0.2838	0.5793	
0.1083	-0.7014	-0.1576	-0.0225	0.6817	0.0565	-0.0215	0.0334	0.0374	
0.1022	0.463	-0.8371	-0.0069	0.2666	-0.017	0.0154	0.0502	-0.0129	
-0.0273	-0.18	-0.2134	-0.1686	-0.313	0.8833	-0.0529	0.0988	-0.0456	

b. K means



step 0

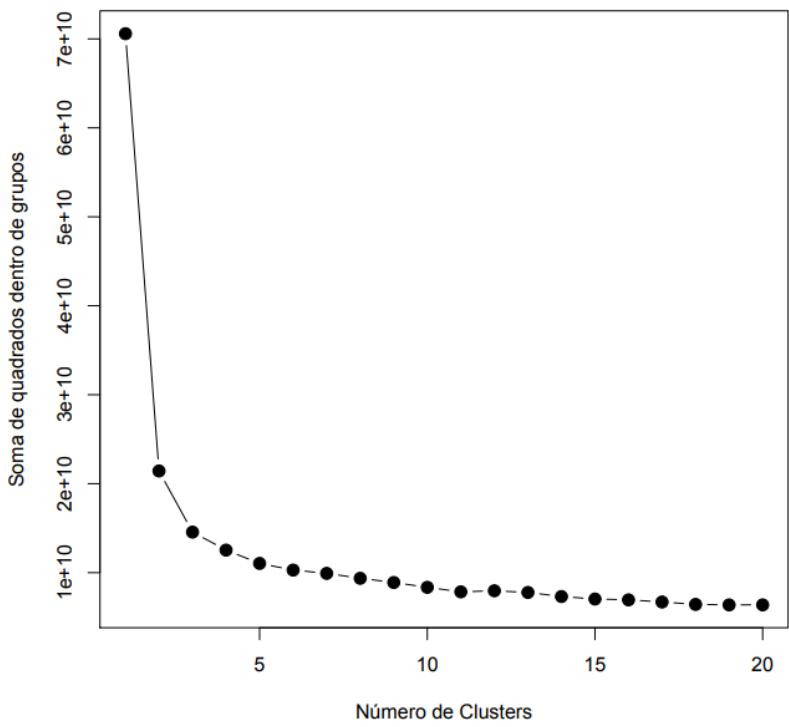


- (1) Determinar o número K, ou seja, o número de grupos nos quais os indivíduos devem ser alocados.
- (2) Inicializar os K centroides aleatoriamente. O centroide k pode assumir um conjunto de dados de um determinado indivíduo tomado aleatoriamente do conjunto original de dados.
- (3) Calcular as distâncias D_{ik} de cada indivíduo em relação a cada centroide e alocar este indivíduo naquele centroide em que a distância é a menor entre todas as distâncias obtidas.
- (3) Atualizar os valores de cada centroide por meio das médias dos indivíduos que foram agrupados, ou seja:

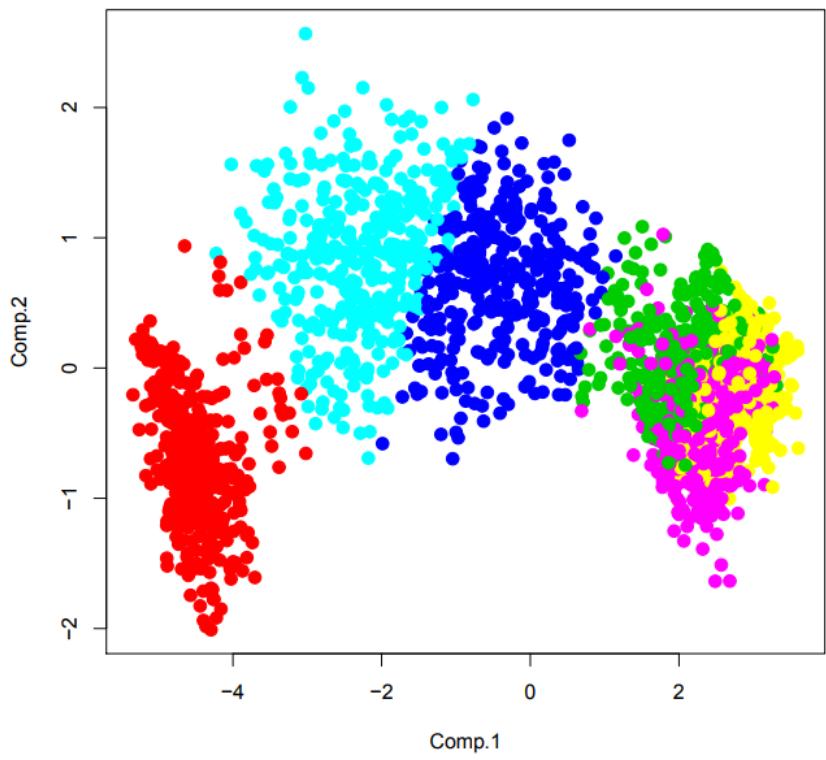
$$c_{kj} = \frac{\sum_{i=1}^{n_k} x_{ij}}{n_k}$$

- (4) Retornar ao passo 3 até que não haja mudança na alocação do indivíduos em relação aos centroides.

Número ótimo de grupos – Método Elbow



K-Means com k grupos



c. Mistura de distribuições

Técnica de estatística multivariada que tem por objetivo criar subgrupos distintos e homogêneos considerando misturas de distribuições Normais Multivariadas.

Ind.	X_1	X_2	...	X_p
1	X_{11}	X_{12}		X_{1p}
2	X_{21}	X_{22}		X_{2p}
...				
n	X_{n1}	X_{n2}		X_{np}

$$\rightarrow \mathbf{X} \sim N_p(\boldsymbol{\mu}, \Sigma).$$

} 1

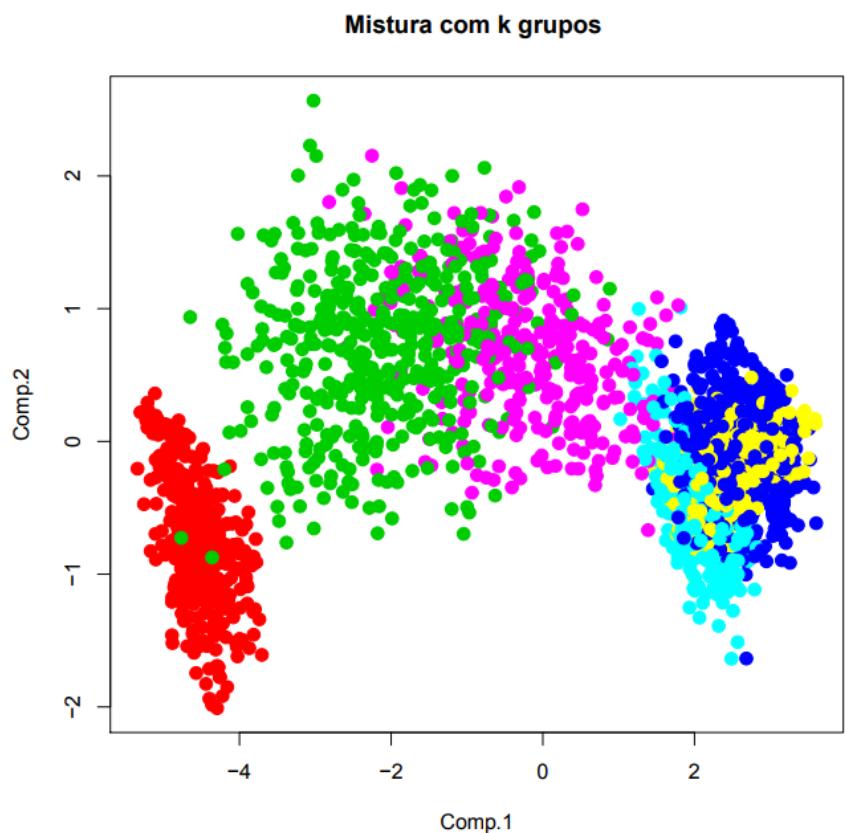
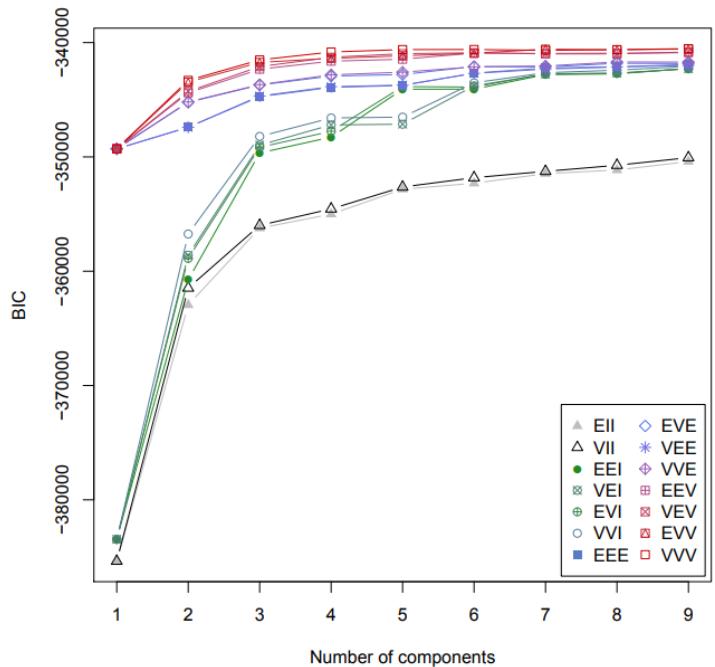
: Particionado em k pop.

} k

$$\rightarrow \mathbf{X}_k \sim N_p(\boldsymbol{\mu}_k, \Sigma_k).$$



c. Mistura de distribuições



Seleção da melhor partição e estrutura de variâncias por BIC

Best BIC values:

BIC VVV,9 EVV,9 EVV,7
 BIC -340528.6 -340558.16101 -340574.89153
 BIC diff 0.0 -29.53637 -46.26689

* Identifique, pelo menor BIC, a melhor partição e melhor estrutura de covariância dos dados

Estudo 2: Análise discriminante – Anderson (ou de Fisher)

Teoria

- Análise Discriminante de Anderson

$$f_j(\tilde{x}) = \frac{1}{2\pi|\Sigma|^{1/2}} e^{-\frac{1}{2}[(\tilde{x}-\mu_j)'\Sigma^{-1}(\tilde{x}-\mu_j)]}$$

$$D_j(\tilde{x}) = -\frac{1}{2} [\ln(2\pi) + \ln|\Sigma_j|] - \frac{1}{2} [(\tilde{x} - \mu_j)' \Sigma_j^{-1} (\tilde{x} - \mu_j)] + \ln(p_j)$$

$$D_j(\tilde{x}) = \kappa_j + \alpha_{j1}x_1 + \alpha_{j2}x_2 + \dots + \alpha_{jv}x_v$$

$$D_1(x) = \ln(p_1) + (x - \frac{1}{2}\mu_1)' \Sigma^{-1} \mu_1$$

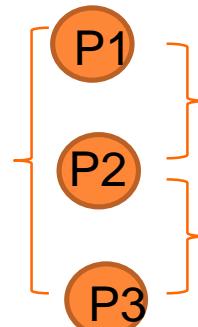
$$D_2(x) = \ln(p_2) + (x - \frac{1}{2}\mu_2)' \Sigma^{-1} \mu_2$$

$$D_3(x) = \ln(p_3) + (x - \frac{1}{2}\mu_3)' \Sigma^{-1} \mu_3$$

- Análise Discriminante de Fisher

$$f_1(\tilde{x}) = \frac{1}{2\pi|\Sigma|^{1/2}} e^{-\frac{1}{2}[(\tilde{x}-\mu_1)'\Sigma^{-1}(\tilde{x}-\mu_1)]}$$

$$\frac{f_1(\tilde{x})}{f_2(\tilde{x})} > 1$$
$$f_2(\tilde{x}) = \frac{1}{2\pi|\Sigma|^{1/2}} e^{-\frac{1}{2}[(\tilde{x}-\mu_2)'\Sigma^{-1}(\tilde{x}-\mu_2)]}$$



Análise Discriminante de Anderson (dados6p_t.dat)

	P1	P2	P3	P4	P5	P6
CONSTANT						
E	-52.9879	-54.547	-52.1194	-15.7154	-48.0737	-13.6593
x1	0.000843	-0.0012	-0.0013	-0.00095	0.000222	-0.00145
x2	0.004472	-0.00465	0.001581	-0.00025	0.003459	-0.00211
x3	0.005219	-0.02164	-0.01194	-0.00569	-0.0021	-0.01125
x4	-0.00074	-0.0031	0.000613	0.000238	0.000802	-0.00115
x5	0.001846	-0.00111	0.00434	0.002444	0.00078	0.001199
x6	0.000248	-0.00886	0.006627	0.004192	0.003782	0.001185
x7	0.001206	0.000775	0.001448	0.000511	0.001584	0.00062
x8	0.003023	0.003442	0.003117	0.001824	0.002854	0.001587
x9	0.008151	0.009868	0.009718	0.005062	0.008429	0.005745

Análise Discriminante de Fisher (dados6p_t.dat)

FUNÇÃO	ES	1,2	1,3	...	5,6
x1	0.002041	0.002146			0.001672
x2	0.009118	0.002891			0.00557
x3	0.026863	0.017164			0.009149
x4	0.002366	-0.00135			0.001948
x5	0.002954	-0.00249			-0.00042
x6	0.009106	-0.00638			0.002597
x7	0.000431	-0.00024			0.000963
x8	-0.00042	-9.4E-05			0.001268
x9	-0.00172	-0.00157			0.002684
PM	-1.55912	0.868451			34.41442

Análise Discriminante

Linear

\$confusion	predicted					
original	1	2	3	4	5	6
1	291	0	0	0	29	0
2	0	320	0	0	0	0
3	0	0	302	0	18	0
4	3	0	10	252	16	39
5	57	0	43	0	220	0
6	0	11	0	48	0	261

Matriz de confusão - Linear/Validação :

	1	2	3	4	5	6
1	60	0	0	0	20	0
2	0	80	0	0	0	0
3	0	0	62	0	18	0
4	1	0	4	56	3	16
5	9	0	10	0	61	0
6	0	2	0	12	0	66

Erros = 274 TEA = 14,27%

Erros = 95 TEA = 19,79%

Quadrática

1.2. Quadrática:

	predicted					
original	1	2	3	4	5	6
1	306	0	0	0	14	0
2	0	320	0	0	0	0
3	0	0	308	2	10	0
4	0	0	2	296	5	17
5	27	0	12	2	279	0
6	0	2	0	28	0	290

Matriz de confusão - Quadrático/Validação :

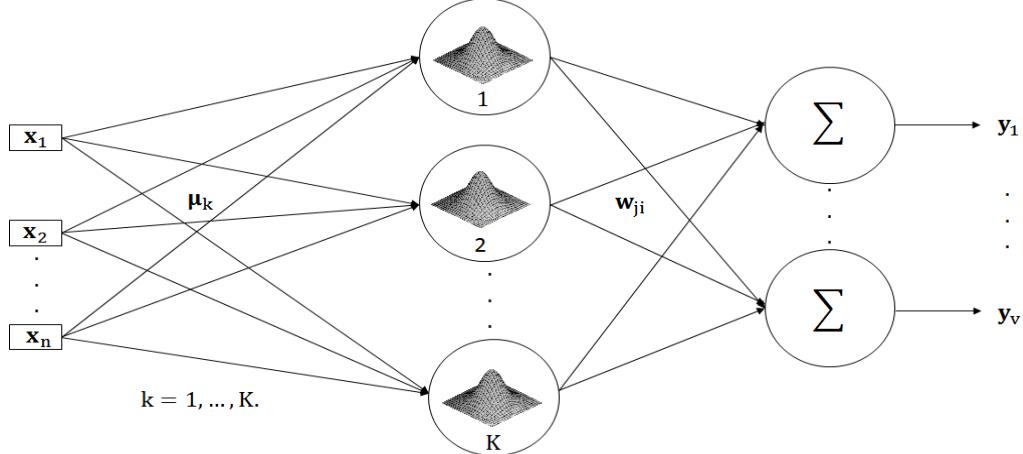
	1	2	3	4	5	6
1	75	0	0	0	5	0
2	0	80	0	0	0	0
3	0	0	69	1	10	0
4	0	0	2	64	2	12
5	1	0	4	0	75	0
6	0	0	0	12	0	68

Erros = 121 TEA = 6,30%

Erros = 49 TEA = 10,21%

Estudo 3: Rede Neural - RBF

A arquitetura das RBF é do tipo *feedforward* com uma camada de entrada, uma intermediária e uma de saída (Figura 12.1).

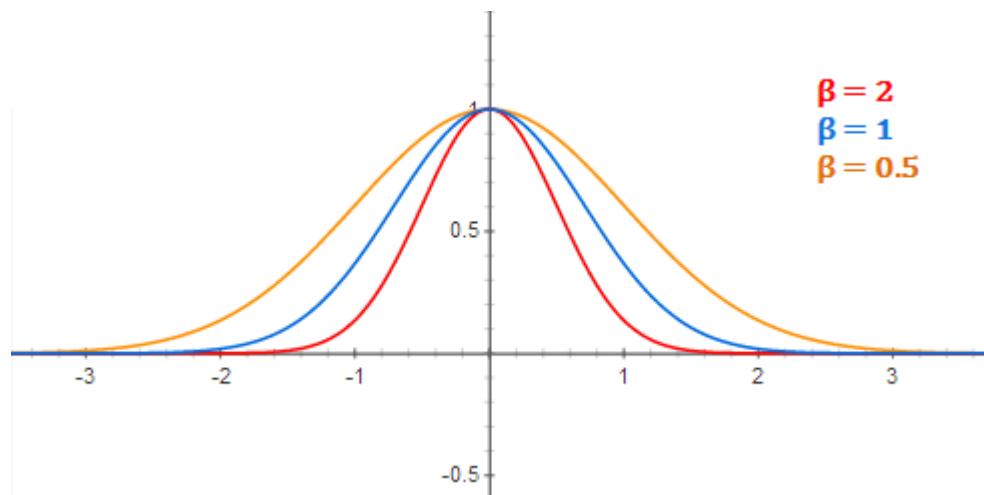


Arquitetura e topologia de uma Rede Funções de Base Radial com n entradas, K neurônios na camada intermediária (RBF) e v saídas.

- primeiro estágio:** realizado treinamento não supervisionado em que o principal objetivo é formar grupos de indivíduos semelhantes, visando a obtenção dos pesos das funções de base radial que compõem os neurônios da camada intermediária.
- segundo estágio:** o treinamento é realizado de maneira semelhante ao executado nas redes PMC, ou seja, supervisionado.

Função empregada na RBF

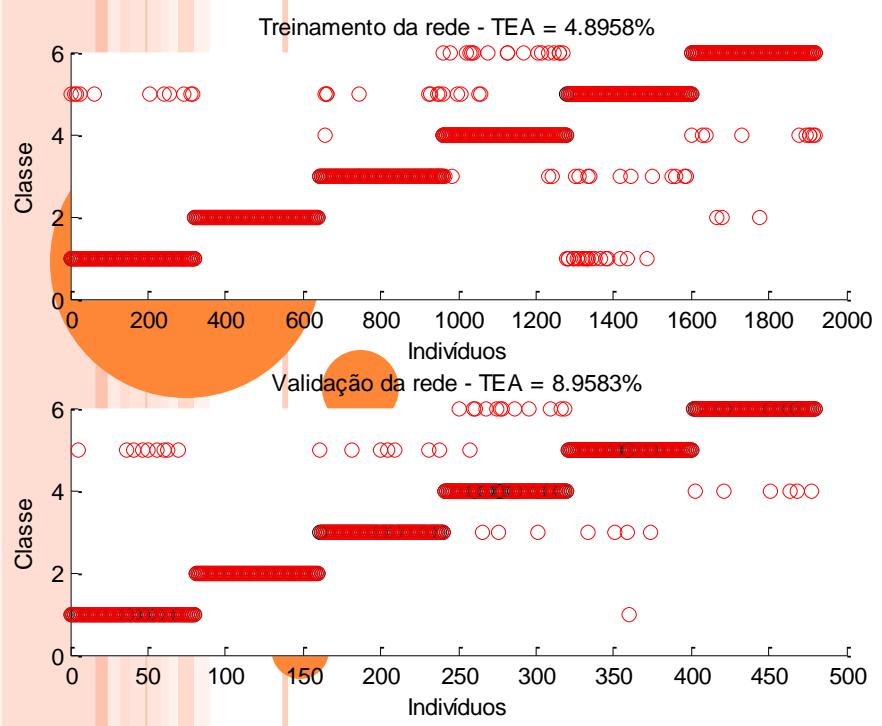
$$\varphi(x) = e^{-\beta\|x-\mu\|^2}$$



Para a função de ativação não estamos diretamente interessados no valor do desvio padrão, então fazemos algumas simplificações:

1. Removemos o coeficiente externo, $1 / (\sigma * \sqrt{2 * \pi})$. Este termo normalmente controla a altura do gaussiano. Aqui, porém, é redundante com os pesos aplicados pelos nós de saída. Durante o treinamento, os nós de saída vai *aprender* o coeficiente correto ou “peso” para aplicar a resposta do neurônio.
2. Substituímos o coeficiente interno, $1 / (2 * \sigma^2)$, com um único parâmetro ‘beta’. Este coeficiente beta controla a largura da curva do sino. Mais uma vez não importa o valor do sigma, apenas o coeficiente que controla a largura da curva do sino. Então, simplificamos a equação substituindo o termo por uma única variável.

Raio ótimo = 0.4 [01 a 2, passo = 0,3]
 NN = 115 [100 a 120, passo 5]
 EQ = 0.01



Raio ótimo : 0.400000
 Número de neurônios ótimo:
 115.000000

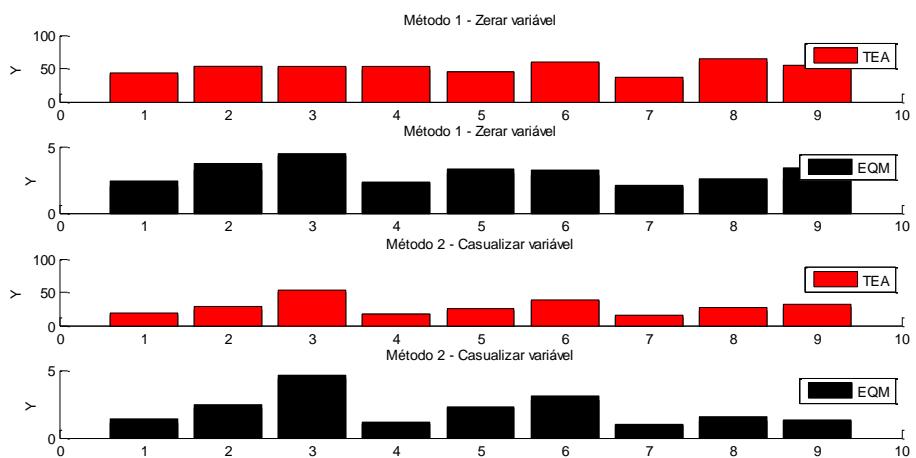


Tabela de confusão - Treinamento =

	Class_P1	Class_P2	Class_P3	Class_P4	Class_P5	Classe_P6
Orig_P1	308.00000	0	0	0	12.00000	0
Orig_P2	0	320.00000	0	0	0	0
Orig_P3	0	0	310.00000	1.00000	9.00000	0
Orig_P4	0	0	4.00000	294.00000	4.00000	18.00
Orig_P5	22.00000	0	11.00000	0	287.00000	0
Orig_P6	0	3.00000	0	10.00000	0	307.00

Tabela de confusão - Validação =

	Class_P1	Class_P2	Class_P3	Class_P4	Class_P5	Classe_P-
Orig_P1	71.00000	0	0	0	9.00000	0
Orig_P2	0	80.00000	0	0	0	0
Orig_P3	0	0	73.00000	0	7.00000	0
Orig_P4	0	0	3.00000	64.00000	1.00000	12.0
Orig_P5	1.00000	0	4.00000	0	75.00000	0
Orig_P6	0	0	0	6.00000	0	74.0

=====

EQM treinamento : 0.413021

EQM validação : 0.583333

TEA treinamento : 4.895833

TEA validação : 8.958333

=====

Estudo 4: Rede Neural - PMC

Eq: 0.02

Número de camadas : 2

Número de neurônios/camada =10

Função de ativação = tansig

Trainamento: trainlm

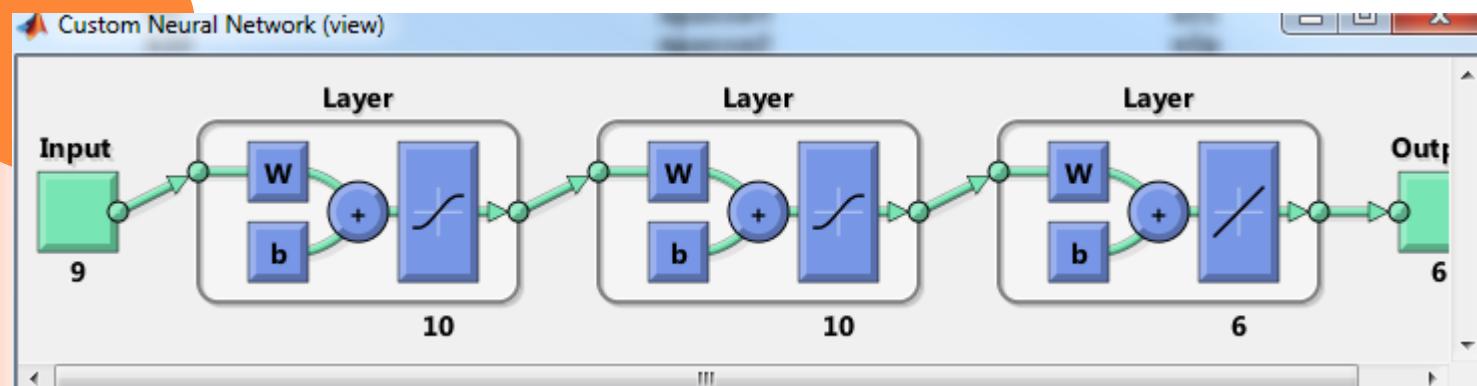
Eq: 43.5

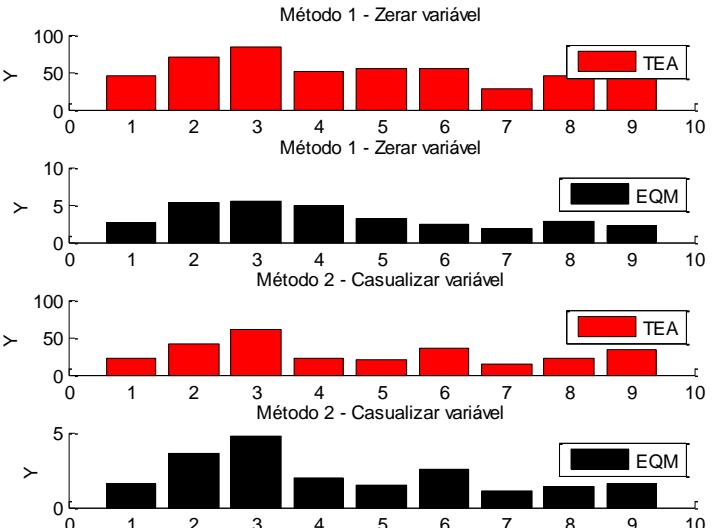
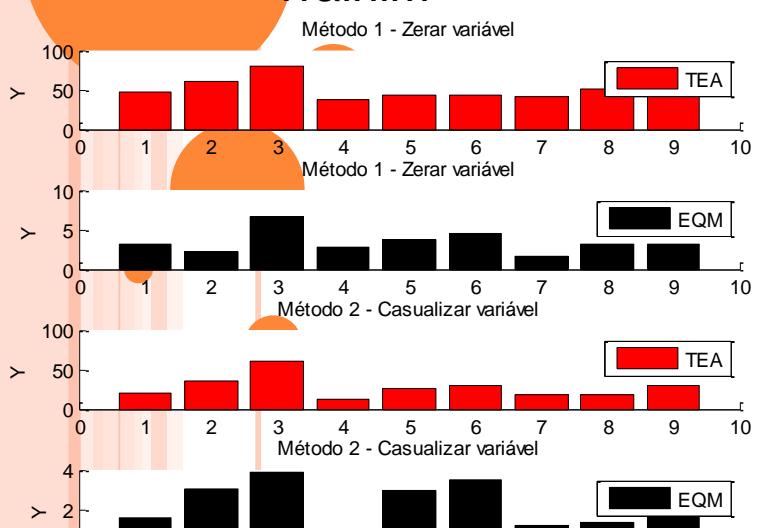
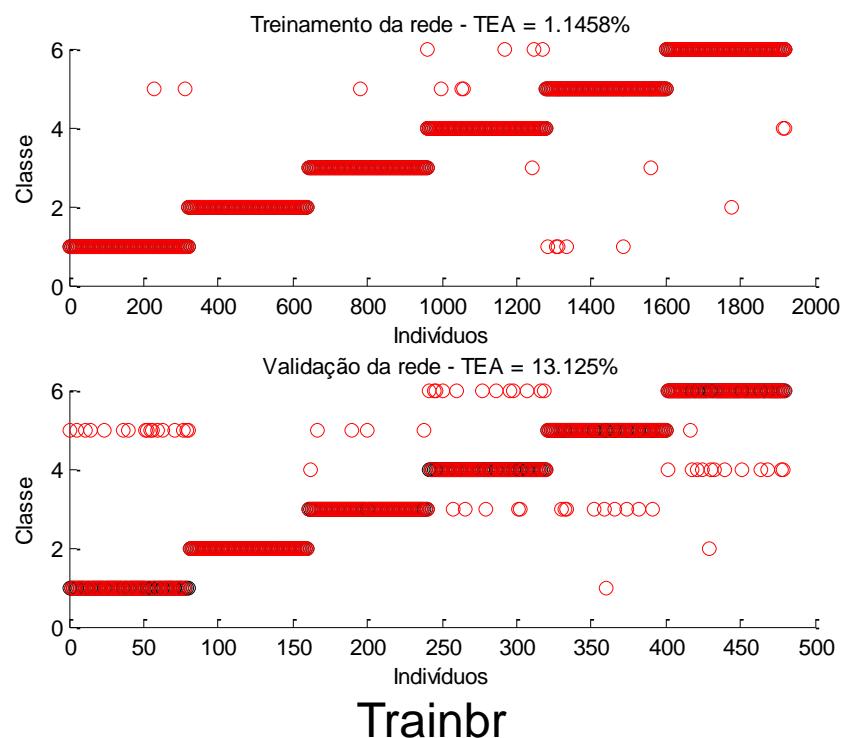
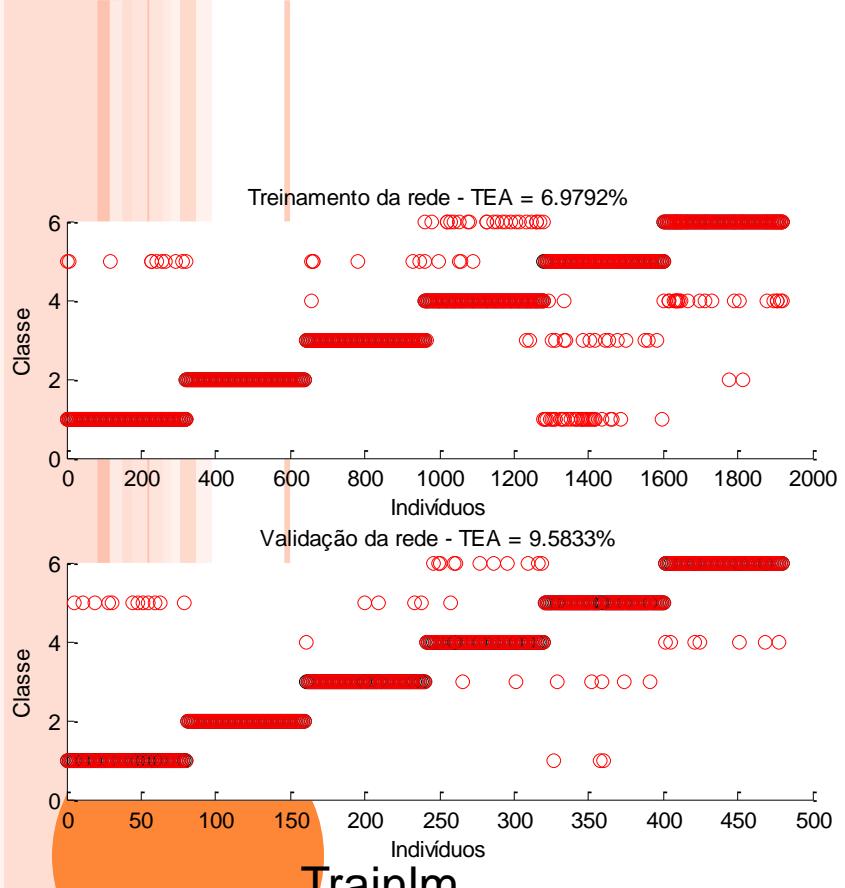
Número de camadas : 2

Número de neurônios/camada =10

Função de ativação = tansig

Trainamento: trainbr



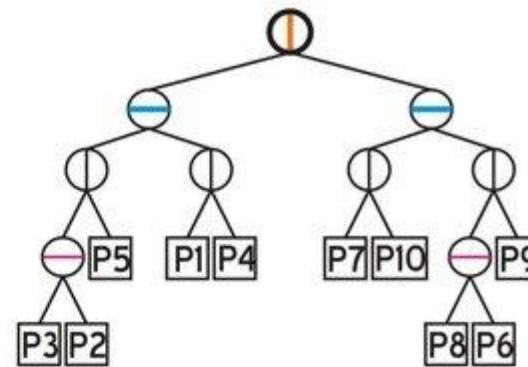
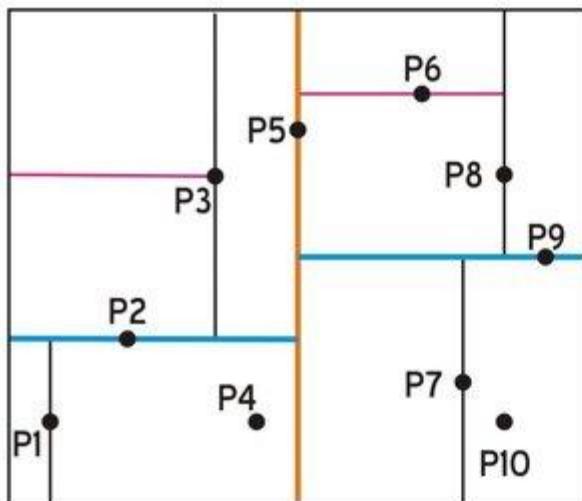


Estudo 5. Árvore de decisão

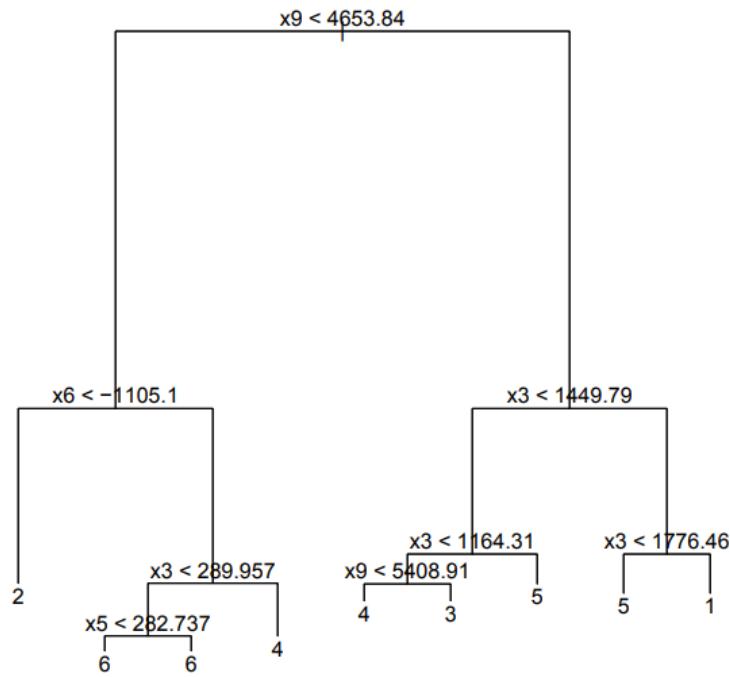
Árvore de decisão é um tipo de algoritmo de aprendizagem supervisionada (com uma variável alvo pré-definida) que é mais utilizada em problemas de classificação e predição.

Ele funciona tanto para as variáveis de entrada e saída categóricas como contínuas.

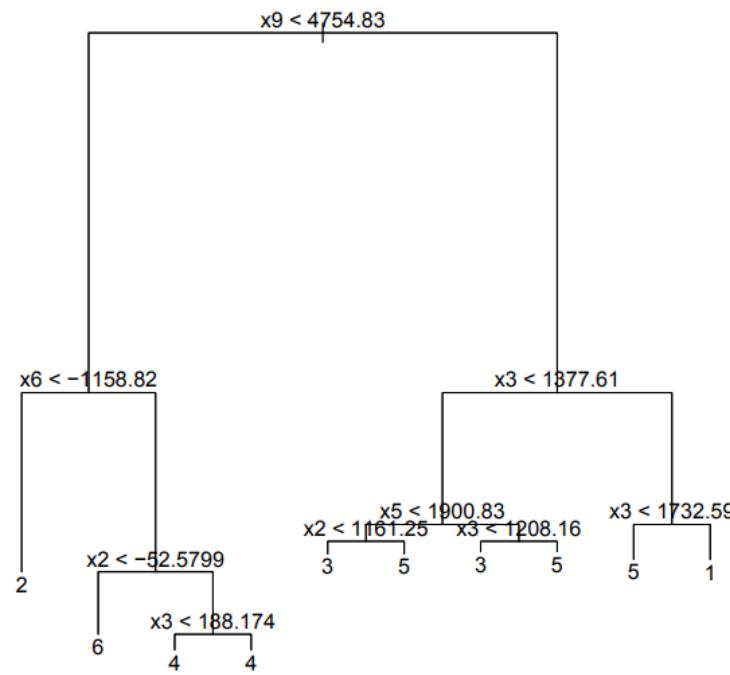
Nesta técnica, dividimos a população ou amostra em dois ou mais conjuntos homogêneos (ou sub populações) com base nos divisores mais significativo / diferenciador das variáveis de entrada.



Árvore de Decisão – Dados de Treinamento



Árvore de Decisão – Dados de Validação



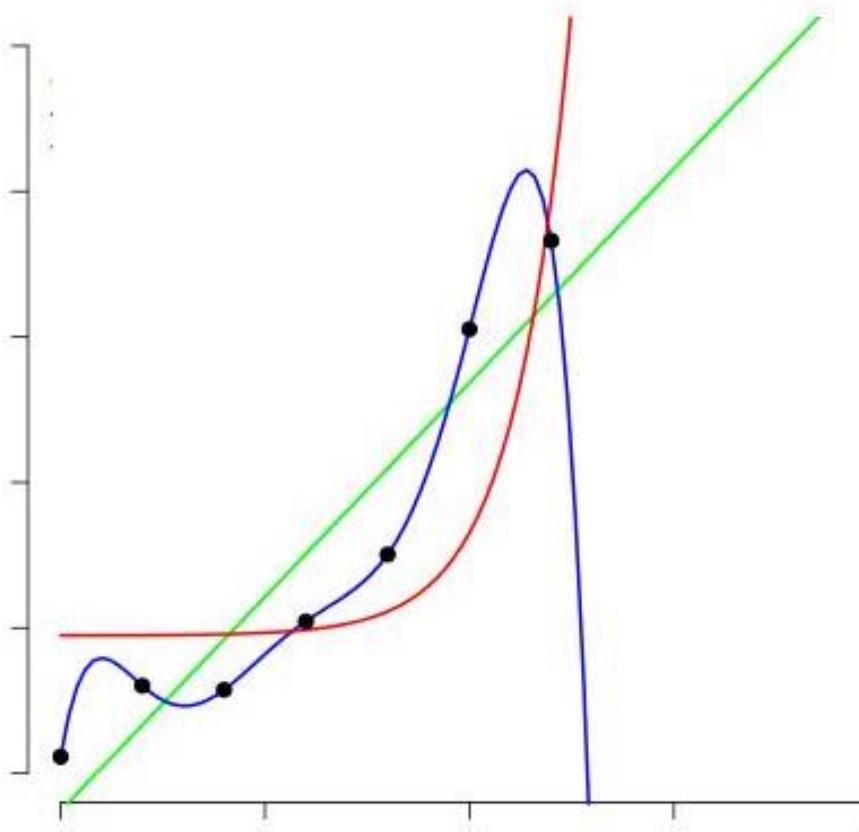
a.	Dados	de	treinamento					
Y.trein								
saídaT	1	2	3	4	5	6		
1	243	0	0	0	47	0		
2	0	256	0	0	0	7		
3	0	0	222	1	11	0		
4	0	0	7	174	10	9		
5	20	0	27	11	187	0		
6	0	0	0	66	0	238		
Acertos :		1320						
Erros :		216						
TEA :		0.140625						

b.	Dados	de	validação					
Y.val								
saídaV	1	2	3	4	5	6		
1	52	0	0	0	14	0		
2	0	64	0	0	0	0		
3	0	0	57	1	1	0		
4	1	0	0	59	0	10		
5	4	0	7	2	50	0		
6	0	0	0	6	0	56		
Acertos :		338						
Erros :		46						
TEA :		0.119792						

Árvore	-	Teste	de	validação			
<hr/>							
Y.val							
saídaVT	1	2	3	4	5	6	
1	49	0	0	0	11	0	
2	0	64	0	0	0	1	
3	0	0	54	0	7	0	
4	1	0	2	47	1	2	
5	7	0	8	1	46	0	
6	0	0	0	20	0	63	
Acertos :		323					
Erros :		61					
TEA :		0.158854					

APLICAÇÃO 2

RNA e ajuste de modelo



Problema: Foram avaliadas 6 variáveis em 200 indivíduos.

Arquivo: c:\dados\plano18_t.dat

c:\dados\plano18_v.dat

Objetivo: Ajustar o modelo $Y_6 = f(Y_1, Y_2, Y_3, Y_4, Y_5)$



a. Estudo da relação por regressão linear múltipla

Entrada de dados

c:\dados\plano18_t.dat

Resultado

ANÁLISE DE REGRESSÃO MÚLTIPLA

FV	GL	SQ	QM	F	PROBABILIDADE
REGRESSÃO	5	7778.9962019	1555.79924038	12.84820076	.00001558
DESVIO	154	18647.98717822	121.09082583		
TOTAL	159	26426.98338012			

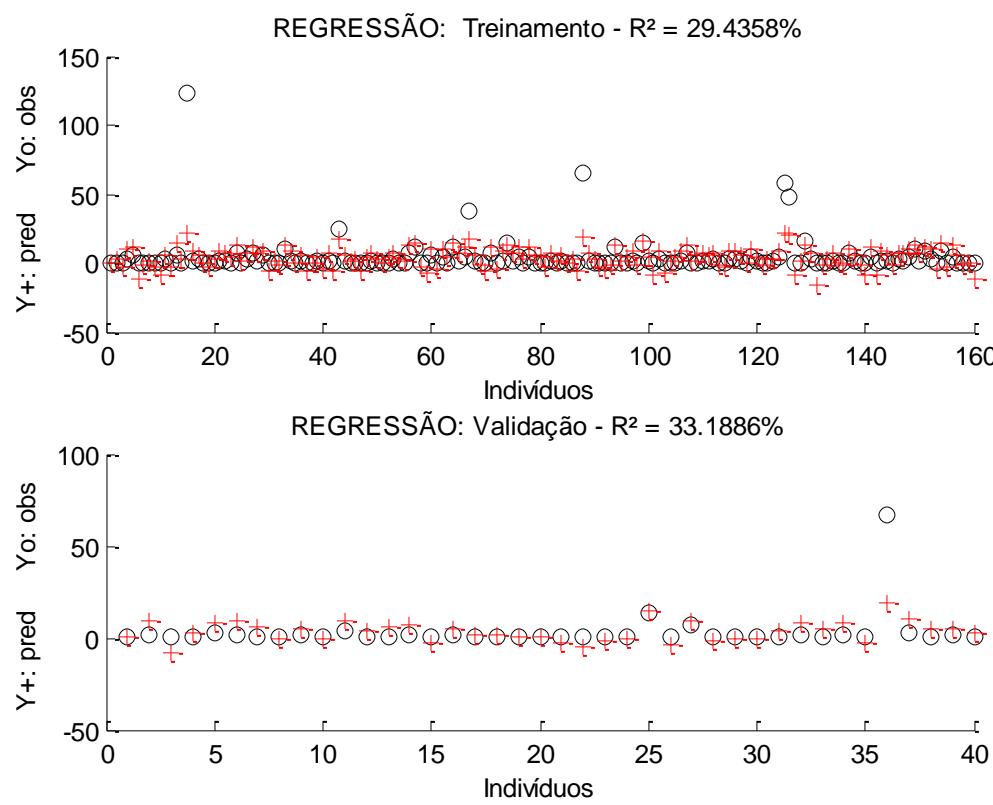
R² (%) **29.43580843**

R² ajustado (%) **27.14476325**

ESTIMATIVAS DOS COEFICIENTES DE REGRESSÃO

NOME	COEFICIENTE (B)	DESVIO	t	PROBAB (*)
x1	.66079837	.08323134	7.93929739	.0
x2	.09121572	.08925359	1.02198378	.30937753
x3	.00698021	.08604604	.08112181	.93315824
x4	.0369665	.08909672	.41490299	.68187284
x5	.05762968	.08082899	.71298281	.48382105
CONSTANTE	-81.9334891			

(*) A PROBABILIDADE FOI OBTIDA PARA O TESTE t BILATERAL



b. Estudo da relação por regressão linear múltipla - Stepwise

FV	GL	SQ	QM	F	PROBAB
REGRESSÃO	1	5029.54304	5029.54304	63.22072	.
x5	1	5029.54304	5029.54304	63.22072	.
DESVIO	158	12569.73721	79.5553		
TOTAL	159	17599.28025	R ² : .2858	R ² Aj : .2813	

c. Análise por rede neural PMC

Topologia

Parâmetros Iniciais

Variável principal (Y) : 6

Épocas : 10000

EQM : 0.01

Camadas Ocultas : 1

Algoritmo Treinamento : trainLM

Earling stoping (1=sim, 2=não): 2

Embaralhar (1=sim, 2=não): 2

Aq treinamento: c:\dados\plano18_t.dat

Aq. Validação c:\dados\plano18_v.dat

Topologia

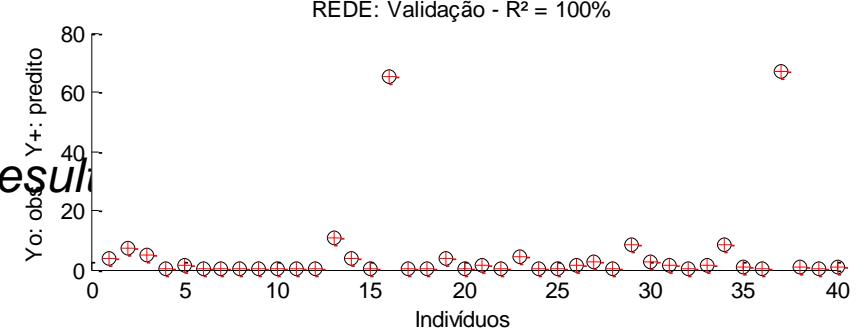
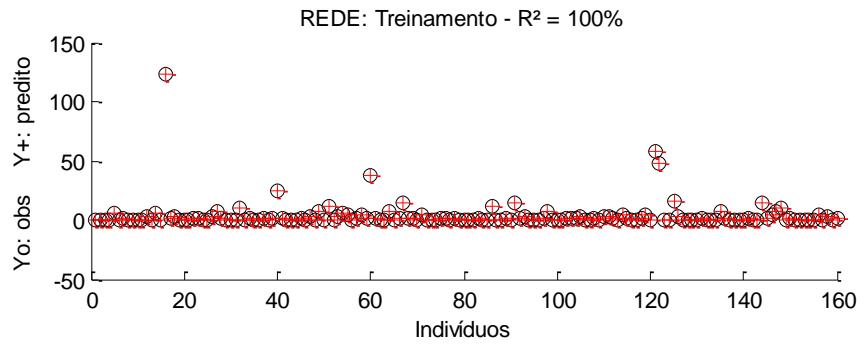
Neurônios na camada 1: **2-2 passo1**

Funçao Atv 1: tansig

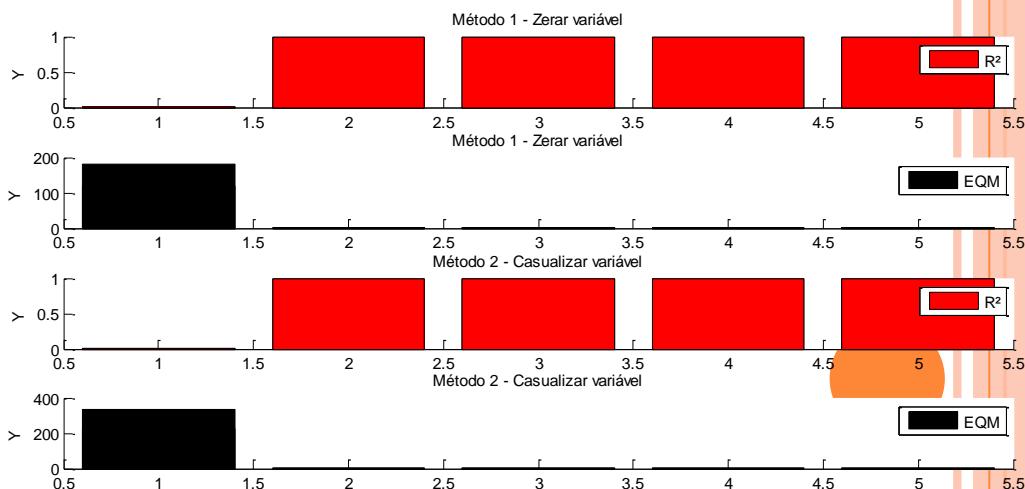
Processamento



Resultados



Resultado – Importância de variáveis

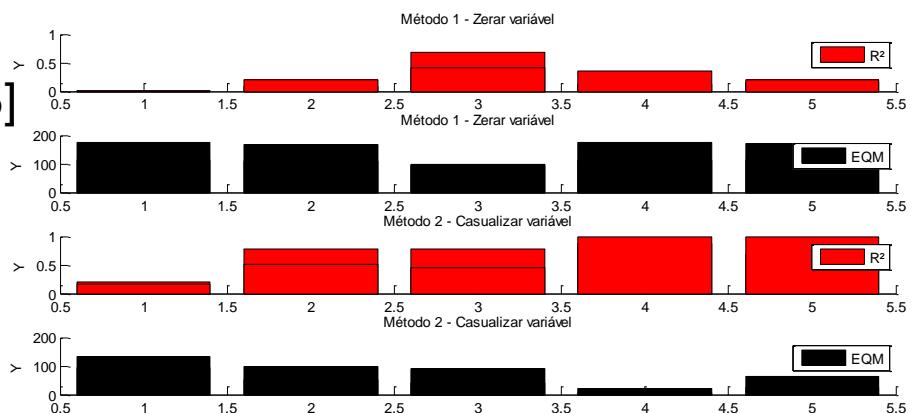
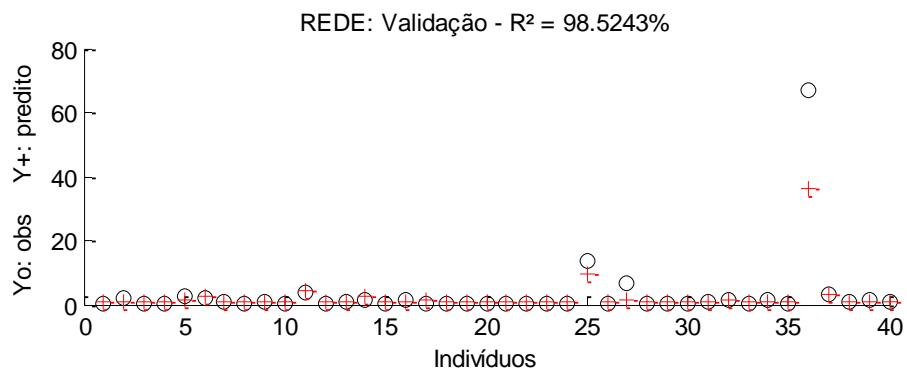
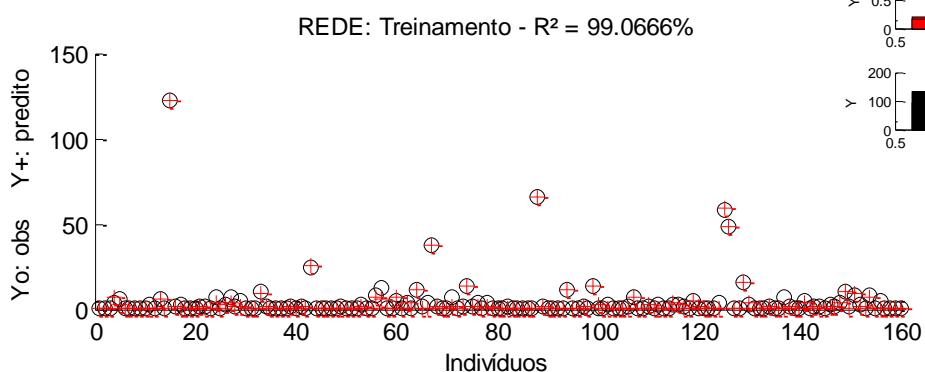


d. Análise por rede neural RBF

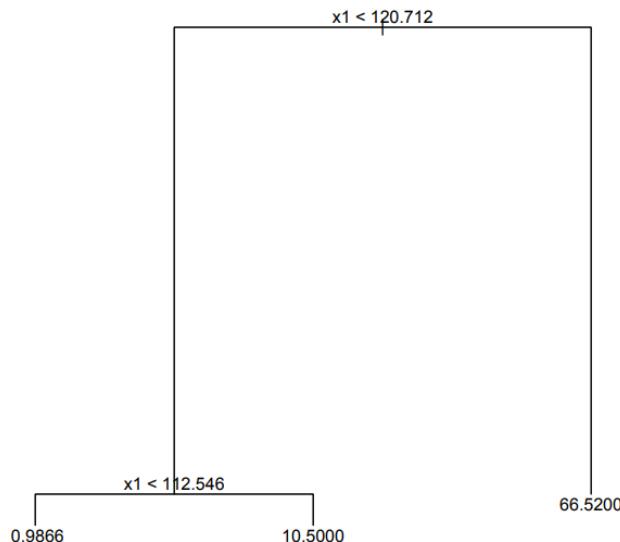
Raio ótimo = 0.05 [0.1 a 1, passo = 0,05]

NN = 12 [12 a 14, passo 2]

EQ = 0.01



e. Análise por rede neural árvore de decisão



Ajuste: Dados apenas de treinamento

Correlação = 0.9485188
R2 = 0.8996878

Árvore - Teste de validação

Correlação = 0.9445806
R2 = 0.8922326



Simulação de Amostras (Vetor de Média μ e Matriz variância Σ)

Sendo X o conjunto representativo das informações de g ($i=1,2\dots g$) indivíduos em relação a v ($j=1,2\dots b$) informações com estrutura de covariância denotada por S, real e simétrica é possível, por meio da decomposição espectral encontrar F de modo que:

$$S^{-1} = FF'$$

atendendo a restrição:

$$F' SF = I$$

É possível obter, por técnicas de componentes principais ou pela simulação, utilizando o teorema de Box-Muller, um novo conjunto de dados, de dimensões equivalentes (pelo menos em relação ao número de colunas, podendo ter o número de linhas reduzido ou ampliado) a de X, denotado por W, que tenha matriz de covariâncias e variâncias igual a I (matriz identidade). Assim, a seguinte transformação pode ser estabelecida:

$$X^* = (F')^{-1}W$$

em que:

$$V(X^*) = (F')^{-1} V(W) (F)^{-1} = (FF')^{-1} = S$$

Assim, o conjunto de dados X^* representa uma réplica do conjunto original de observações com as mesmas propriedades de variância e covariância.