FIAP GRADUAÇÃO

FIAP

ENGENHARIA DE COMPUTAÇÃO

Inteligência Artificial e Computacional

PROF. ANTONIO SELVATICI

Versão 4 – Agosto/2018

SHORT BIO

É engenheiro eletrônico formado pelo Instituto Tecnológico de Aeronáutica (ITA), com mestrado e doutorado pela Escola Politécnica (USP), e passagem pela Georgia Institute of Technology em Atlanta (EUA). Desde 2002, atua na indústria em projetos nas áreas de robótica, visão computacional e internet das coisas, aliando teoria e prática no desenvolvimento de soluções baseadas em Machine Learning, processamento paralelo e modelos probabilísticos. Desenvolveu projetos para Avibrás, IPT e Systax.

PROF. ANTONIO SELVATICI

profantonio.selvatici@fiap.com.br

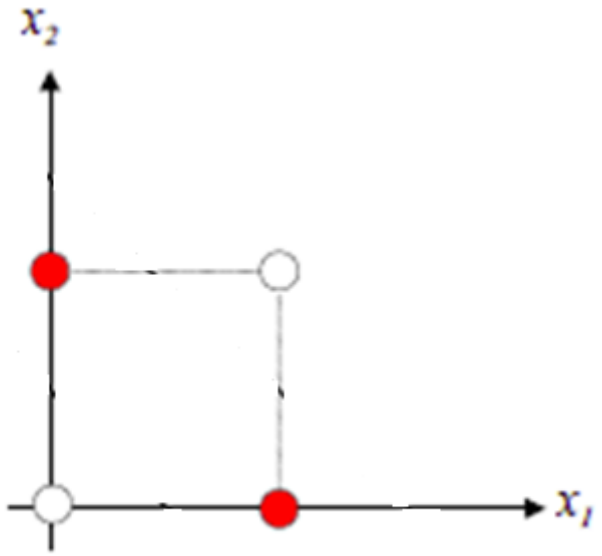
2. MACHINE LEARNING

Problema da classificação XOR através do Perceptron

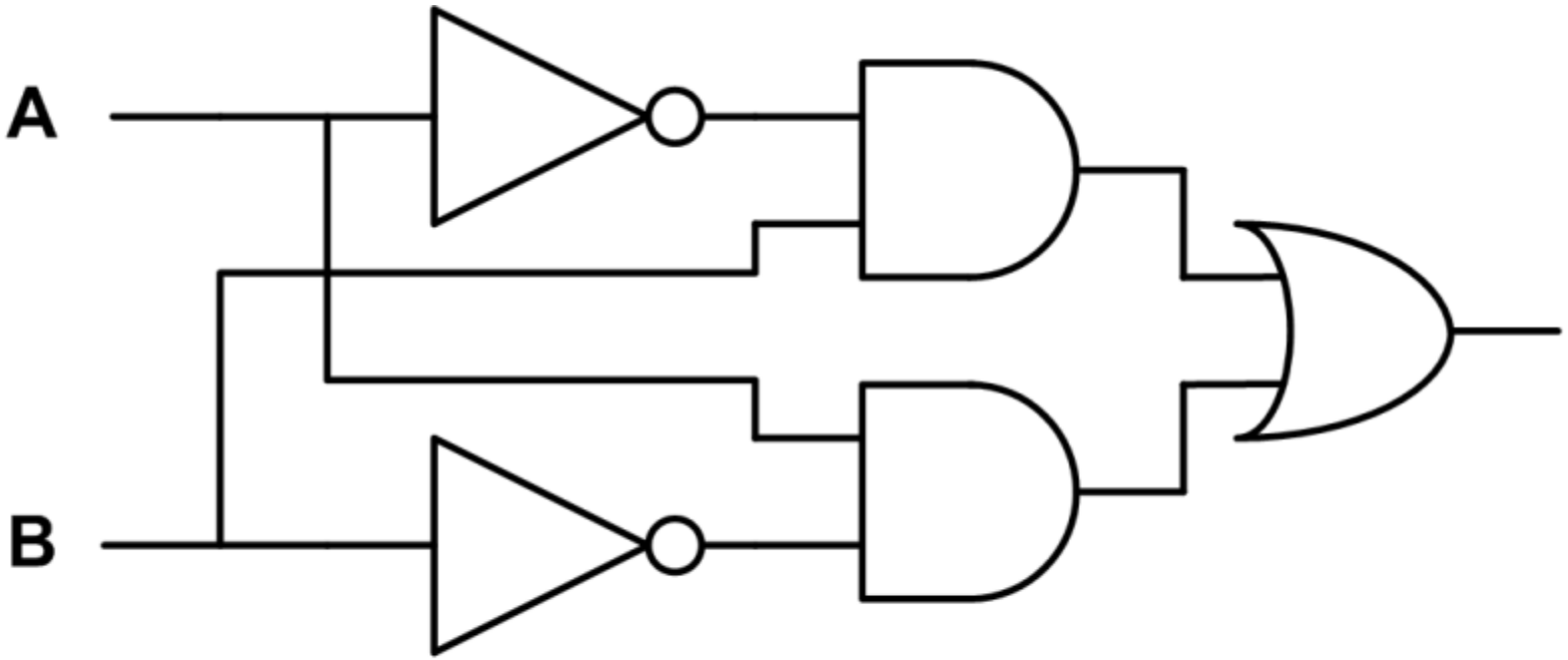
- Como encontrar uma reta que separe os pontos brancos dos pontos vermelhos?

- Tabela verdade XOR:

x1	x2	x1 XOR x2
0	0	0
1	0	1
0	1	1
1	1	0



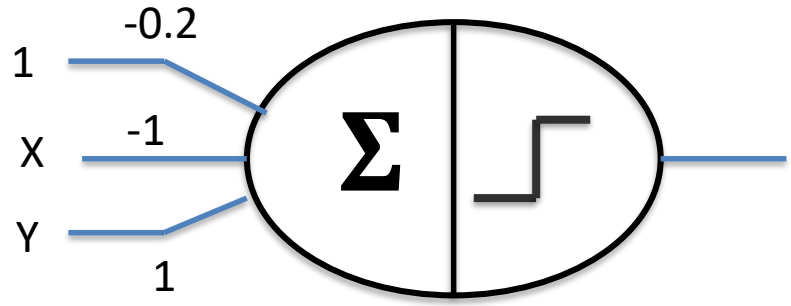
Solução com portas lógicas

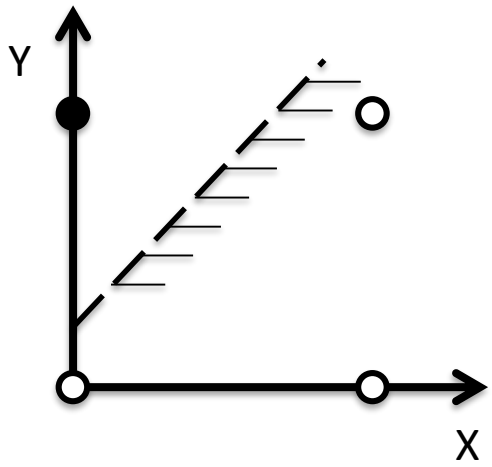


A xor B = A'B + AB'

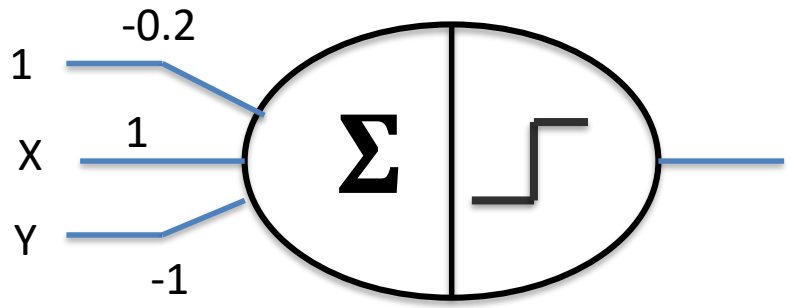
Portas lógicas com Perceptrons

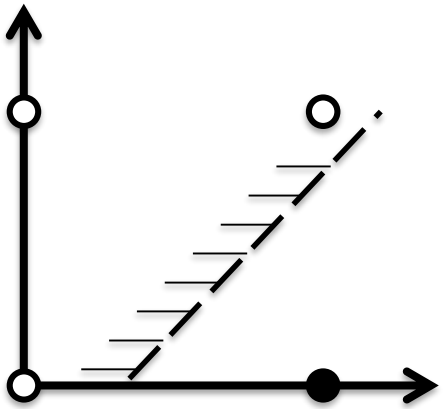
Porta X' AND Y



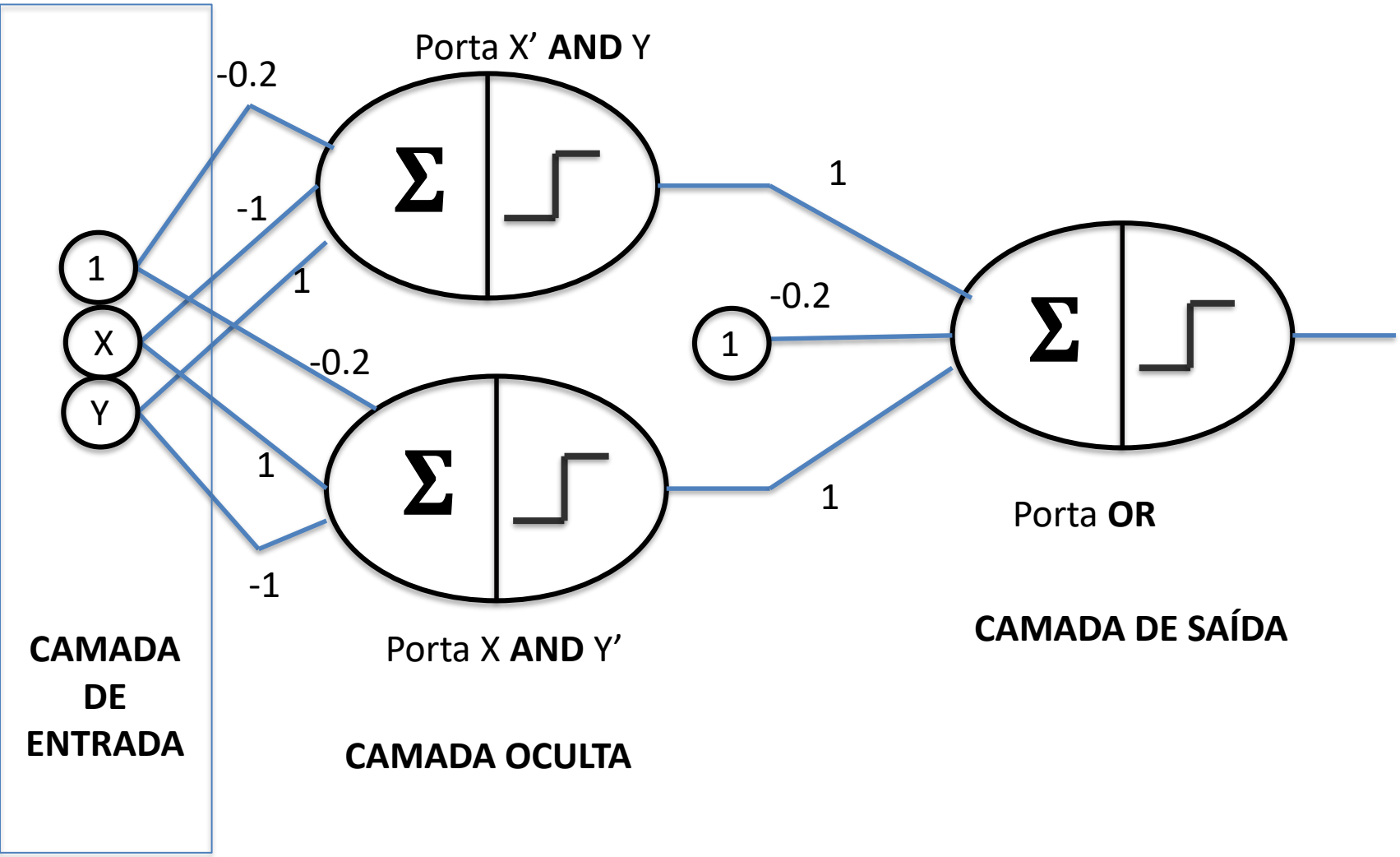


Porta X AND Y'

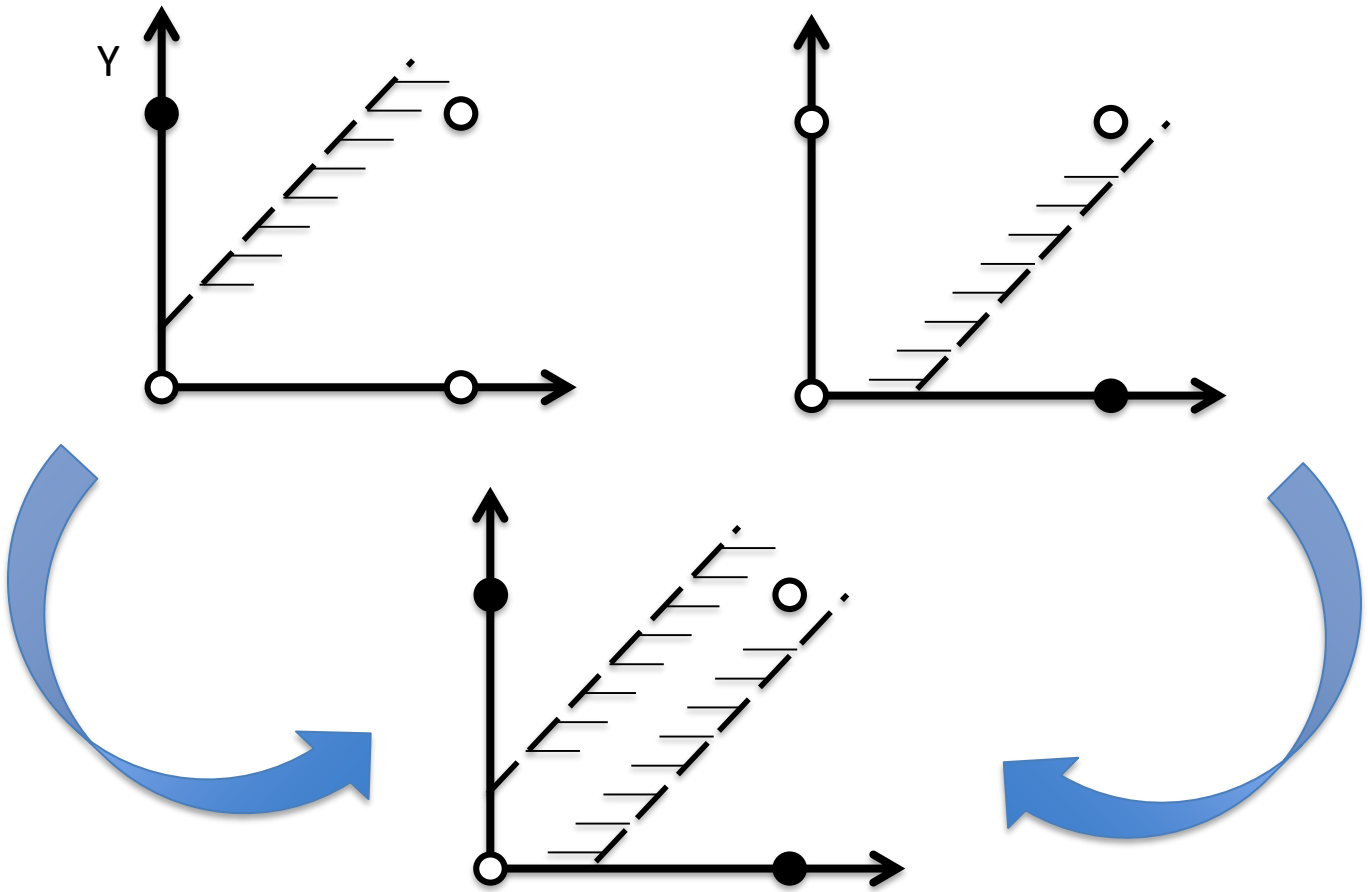




Acrescentando uma terceira camada é possível!



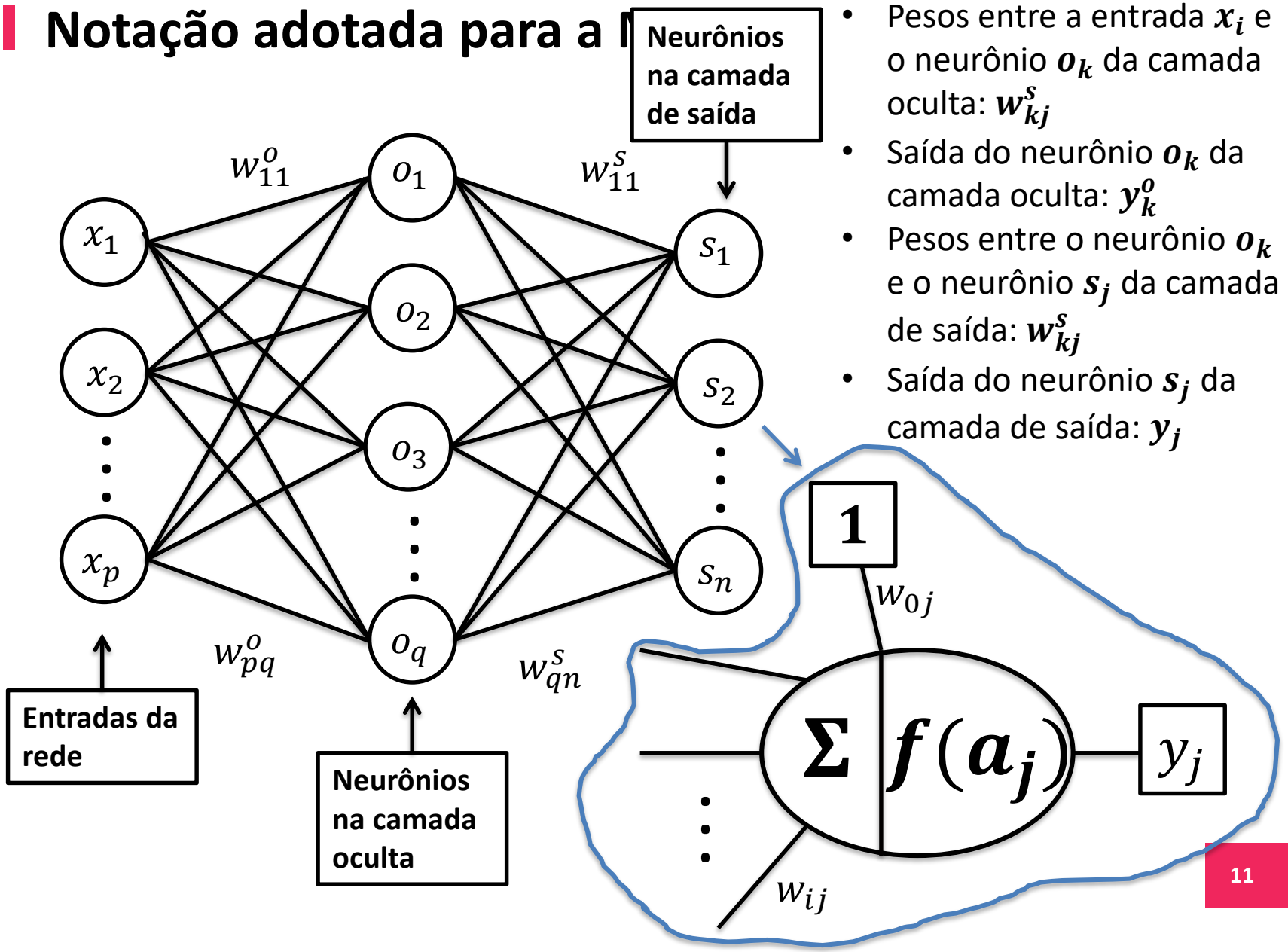
Superfície de separação



Multilayer Perceptron (MLP)

- O acréscimo de uma nova camada de neurônios, denominada **camada oculta**, permite criar superfícies de separação não lineares, permitindo a classificação de classes *não-linearmente separáveis*
- A rede MLP é considerada uma rede do tipo *feed-forward*, já que as saídas dos neurônios das camadas posteriores dependem apenas dos neurônios das camadas anteriores
- Em uma rede MLP, não há regra para o número de neurônios a ser usado na camada oculta, e nem há limites para o número de camadas ocultas a serem usadas
- Aparentemente, um bom chute inicial é considerar o dobro de neurônios na camada oculta com relação ao tamanho da entrada
- É conhecido que com uma única camada oculta com um número suficientemente grande de nós é possível representar qualquer função contínua, e por isso essa estrutura é conhecida como aproximador universal

Notação adotada para a R

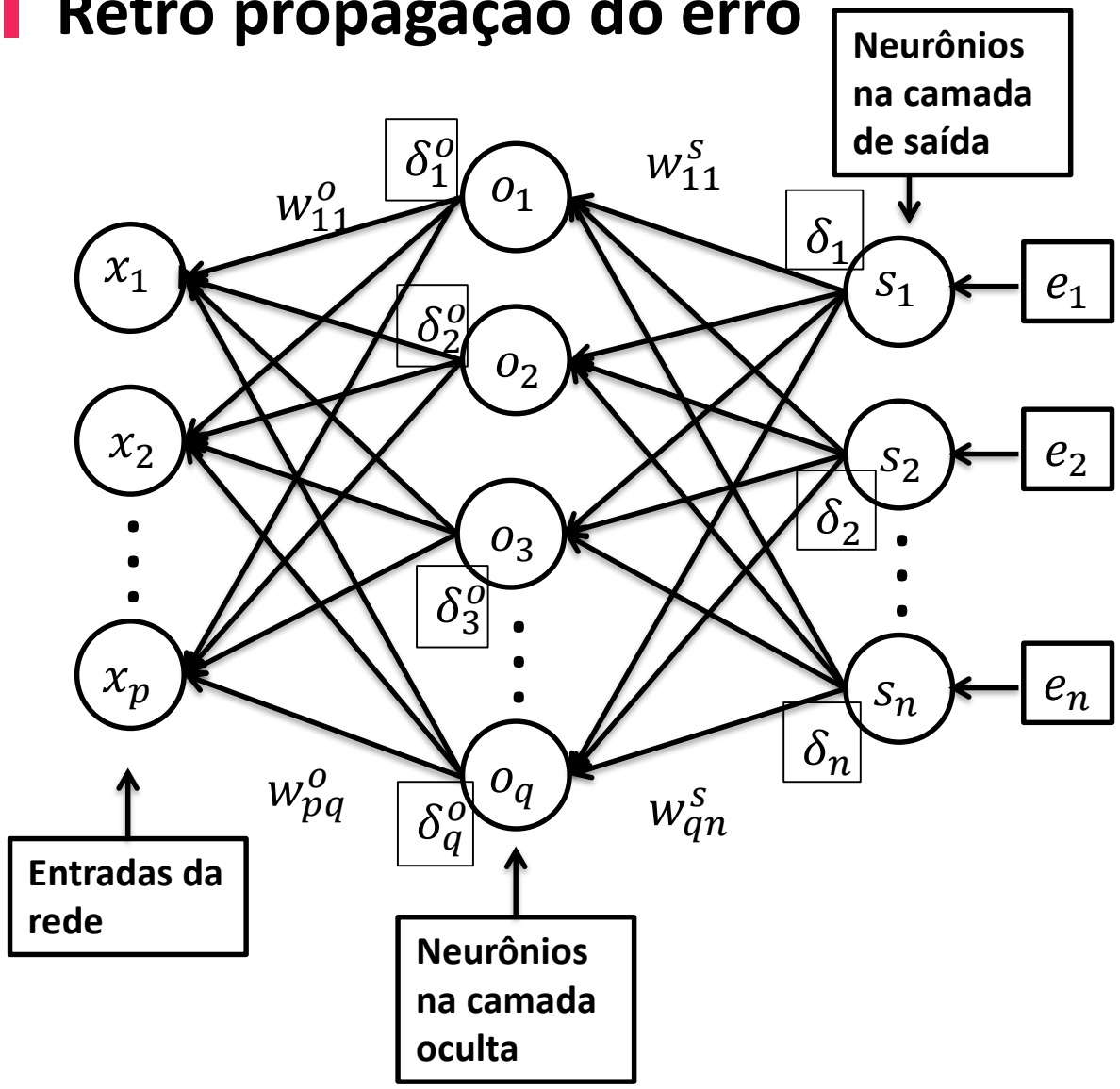


- Pesos entre a entrada x_i e o neurônio o_k da camada oculta: w_{kj}^{s}
- Saída do neurônio o_k da camada oculta: y_k^{o}
- Pesos entre o neurônio o_k e o neurônio s_j da camada de saída: w_{kj}^{s}
- Saída do neurônio s_j da camada de saída: y_j

Aprendizado do Perceptron Multicamadas

- Em 1974, Paul Werbos foi o primeiro a definir um método geral e eficaz de aprendizado da rede Perceptron Multicamadas, denominado **backpropagation**.
- O backpropagation teve uso restrito até que em 1986, David Rumelhart, Geoffrey Hinton e Ronald Williams publicaram um trabalho demonstrando a capacidade do backpropagation em gerar uma representação interna interessante em termos dos pesos do neurônio
  - Diferentemente do caso linear (Perceptron simples), o backpropagation não tem garantia de convergência para a melhor superfície de separação entre as classes, mas em geral consegue produzir resultados interessantes
  - Embora tenha perdido sua importância nos anos 2000 em favor de outros algoritmos de aprendizado de redes neurais, a partir dos anos 2010, com o uso de processamento intensivo e paralelo, o backpropagation demonstrou sua capacidade de treinar redes neurais de vária camadas e milhares de neurônios, resultando no que hoje muitos chamam de “deep-learning”
- O objetivo dos algoritmos de aprendizado é minimizar a soma dos erros quadráticos da saída da rede neural, ou seja, para cada neurônio de saída

Retro propagação do erro



$$\Delta w_{ij} = \eta . \delta_j . x_i$$

δ_j é o erro de saída da rede neural propagado para a entrada de cada neurônio j

O algoritmo SGD aplicado às MLPs

- SGD significa *Stochastic Gradient Descent*
- O objetivo do SGD é encontrar o conjunto de parâmetros (pesos) que reduzem uma função de custo através de um método de busca denominado "Descida pelo Gradiente"
- Em primeiro lugar, a função de custo tradicionalmente usada pela MLP é a média da soma dos erros de cada neurônio de saída ao longo das diferentes entradas:

$$\mathcal{L}(W) = \frac{1}{2m}\left[\Sigma_{r=1}^{m}\Sigma_{j=1}^{n}\left(d_j(X_r) - y_j(X_r, W)\right)^2\right]$$

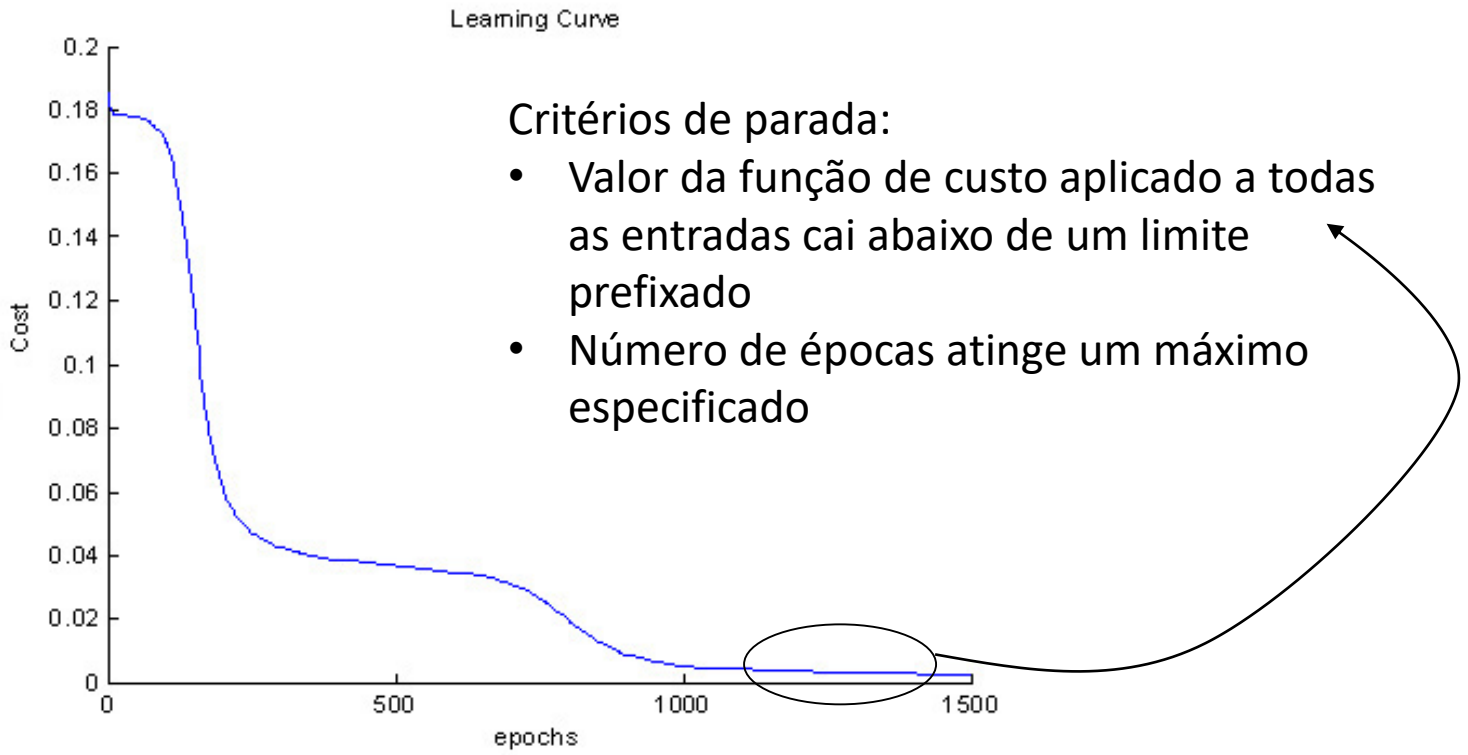
  - **m** – número de entradas sendo processadas de cada vez
  - **n** – número de neurônios na camada de saída
- A quantidade **m** de entradas sendo considerada no custo define quantas entradas serão processadas a cada passo do algoritmo para definir o ΔW, ou seja, as modificações nos pesos sinápticos

Cenários do SGD

- Ainda com relação ao número de entradas a serem processadas, podemos pensar em 3 cenários diferentes
  1. O número **m** corresponde ao total de entradas disponíveis para o treinamento da rede neural.
  2. O número **m** corresponde a uma pequena fração das entradas disponíveis para o treinamento da rede neural, o que chamamos de "*batch size*"
  3. **m** = 1
- Embora não garanta a convergência do algoritmo para o melhor conjunto de pesos, o cenário 1 proporciona as melhores modificações nos pesos, no sentido de levar os pesos a convergirem mais rapidamente, resultando no algoritmo Gradient Descent original.
- No entanto, nos casos onde há milhares ou milhões de amostras de treinamento, possivelmente de alta dimensionalidade, carregar e processar essas amostras na memória do computador muitas vezes não é possível
  - Os cenários 2 e 3 são usados nesses casos, embora devamos notar que a convergência do algoritmo fica prejudicada.
  - O cenário 3 dá origem ao algoritmo de **backpropagation** original, onde os pesos são atualizados após analisar cada amostra de treinamento
- https://www.youtube.com/watch?v=hMLUgM6kTp8

Curva de aprendizado

- A “curva de aprendizado” indica o valor da função de custo aplicada a todos os dados de entrada ao final de cada **época**, ou seja, cada vez que repetimos o uso das mesmas amostras
- Durante o aprendizado, os algoritmos reutilizam as amostras de treinamento várias vezes para tentar diminuir ou estabilizar em um valo mínimo o erro da saída
- A cada época precisamos randomizar a ordem em que as amostras são avaliadas



Critérios de parada:

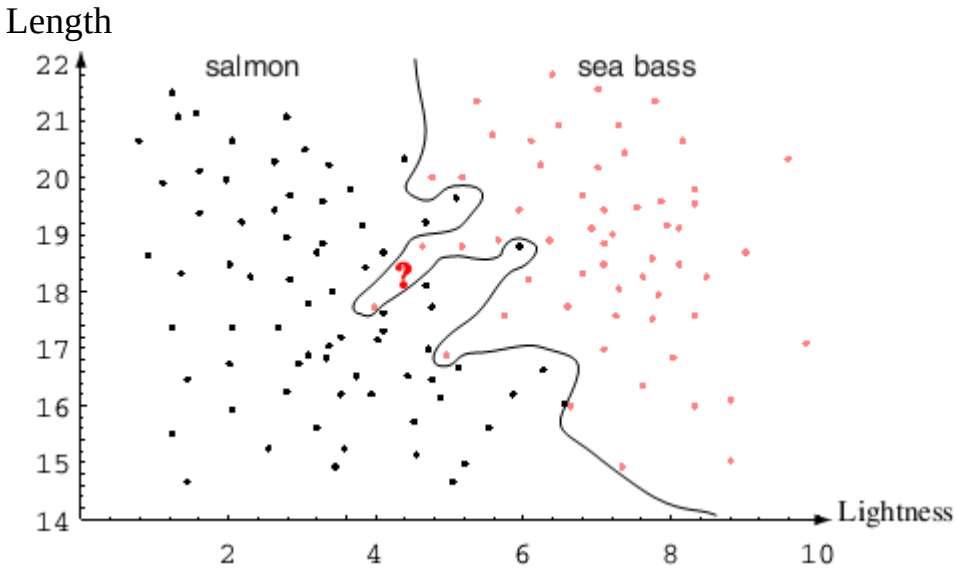
- Valor da função de custo aplicado a todas as entradas cai abaixo de um limite prefixado
- Número de épocas atinge um máximo especificado

Exercício

- Use a rede neural do slide 8, substituindo a função de ativação de step function para logística, e calcule a sua saída para as seguintes entradas:
  - x = 0,5 e y = 0,5
  - x = 0,2 e y = 1,2
- No site `playground.tensorflow.org`, use a rede neural acima para verificar a capacidade de separação de classes dos dados rotulados de "Exclusive OR"
  - Qual é a configuração da rede que, após o treinamento, consegue de fato fazer essa separação?

Generalização do aprendizado

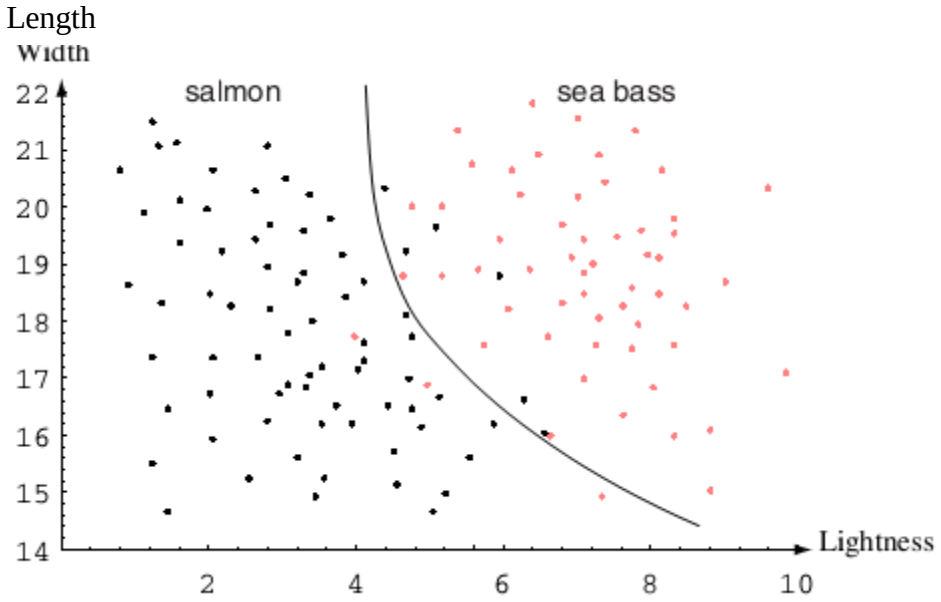
- Um classificador não linear pode, a princípio, usar os dados de treinamento para definir uma superfície de separação entre classes de forma que, quando fazemos a classificação desses dados, o resultado é 100% correto, como no exemplo abaixo.



- No entanto, a classificação do novo vetor de atributos marcado como ? pode não ser realizada corretamente. Uma análise visual do gráfico indica, inclusive uma maior probabilidade desse vetor corresponder a um salmão em vez de um badejo. A esse problema no aprendizado de máquina damos o nome de **overfitting**.

8 Generalização do aprendizado

- Seja a superfície de separação de classes representada abaixo:



- Embora nem todos os dados de aprendizado tenham sido classificados corretamente, a linha desenhada no gráfico separa duas regiões que claramente possuem grande probabilidade de determinar a classe do vetor de atributos de forma correta. Dessa forma, podemos dizer que esse classificador tem melhor capacidade de generalização do aprendizado do que o classificador anterior.

REFERÊNCIAS

- Stuart Russel & Peter Norvig. Inteligência Artificial – tradução da 2ª ed. Editora Campus, 2004
- Duda, Hart & Stork. Pattern Classification, 2nd. Ed., 2000
- Andre P. L. Carvalho. Redes Neurais Artificiais. http://www.icmc.usp.br/~andre/research/neural/
- Cassia Y. Tatibana & Deisi Y. Kaetsu. Redes Neurais. http://www.din.uem.br/ia/neurais/

FIAP