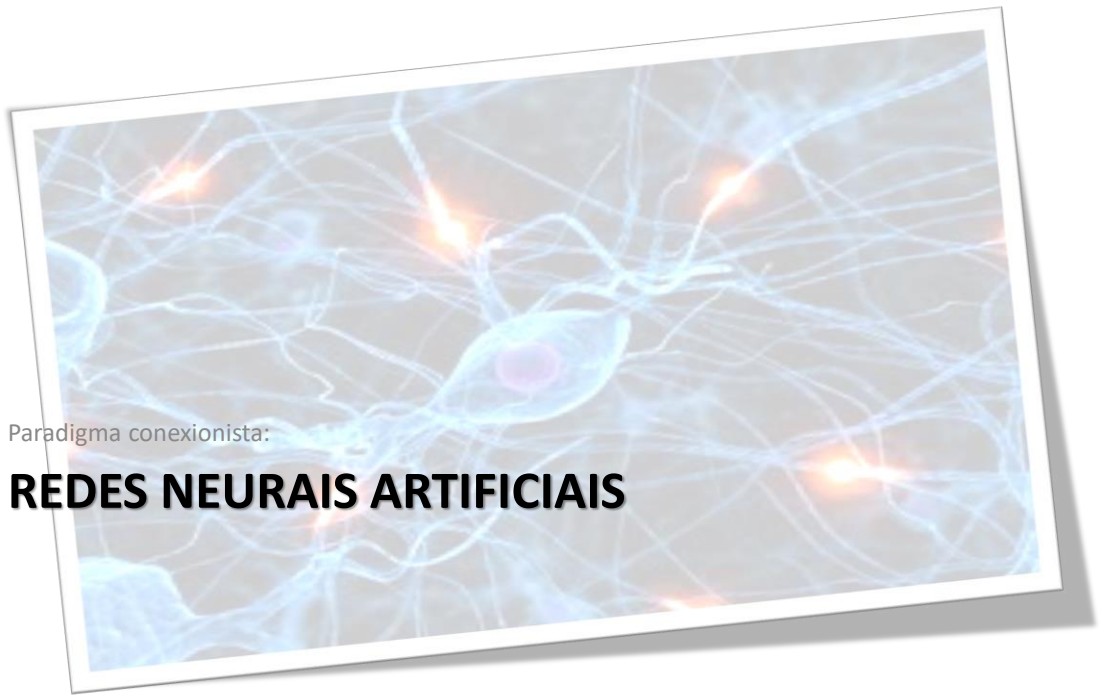


Paradigma conexionista:

REDES NEURAIS ARTIFICIAIS

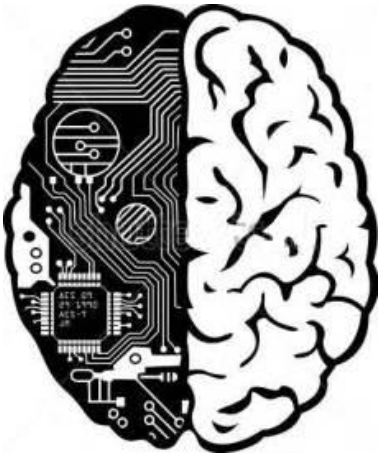
Redes Neurais

- Modelos inspirados no cérebro humano, criadas em analogia a sistemas neurais biológicos, que são capazes de aprendizagem.
  - Compostas por várias unidades de processamento (“neurônios”)
  - Interligadas por um grande número de conexões (“sinapses”)
- Criadas com o objetivo de entender sistemas neurais biológicos através de modelagem computacional.
  - Entretanto existe uma grande divergência entre os modelos biológicos neurais estudados em neurociência e as redes neurais usadas em aprendizagem de máquina.

Redes Neurais

- O caráter “distribuído” das representações neurais permite robustez e degradação suave.
- Comportamento inteligente é uma propriedade “emergente” de um grande número de unidades simples ao contrário do que acontece com regras e algoritmos simbólicos.

Cérebro Humano vs. Computadores



- Neurônios “ligam” e “desligam” em alguns milissegundos, enquanto o hardware atual faz essa mesma operação em nanossegundos.
- Entretanto, os sistemas neurais biológicos realizam tarefas cognitivas complexas (visão, reconhecimento de voz) em décimos de segundo.
- Sistema neural utiliza um “paralelismo massivo”
  - Cérebro humano tem 10^{11} neurônios com uma média de 10^4 conexões cada.
  - Lentidão compensada por grande número de neurônios massivamente conectados.

Redes Neurais Artificiais

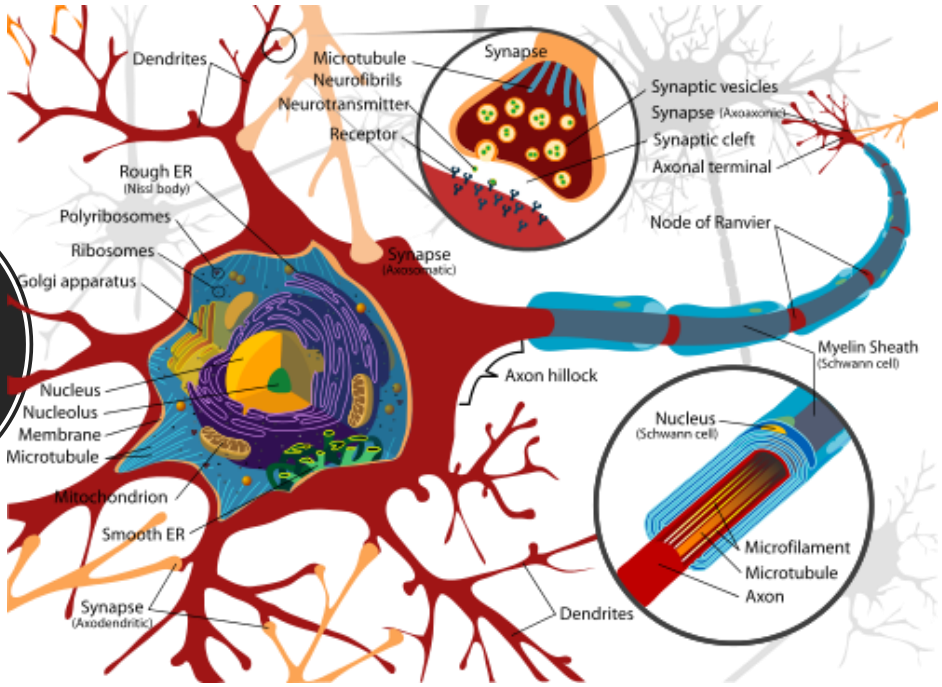
- Redes Neurais Artificiais (RNAs) são tentativas de produzir sistemas de aprendizado biologicamente realistas.
  - São baseadas em modelos abstratos de como pensamos que o cérebro (e os neurônios) funcionam
  - RNAs aprendem por exemplo
  - RNA = arquitetura (modelo/topologia) + processo de aprendizado
- São sistemas "celulares" que podem adquirir, armazenar e utilizar aprendizado por experiência.

Processo de Aprendizagem em Redes Neurais

- Abordagem baseada numa adaptação do funcionamento de sistemas neurais biológicos.
  - Perceptron: Algoritmo inicial pra aprendizagem de redes neurais simples (uma camada) desenvolvido nos anos 50.
  - Retropropagação (*backpropagation*): Algoritmo mais complexo para aprendizagem de redes neurais de múltiplas camadas desenvolvido nos anos 80.

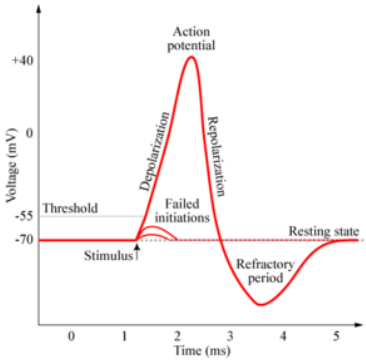
6

Neurônios "Naturais"

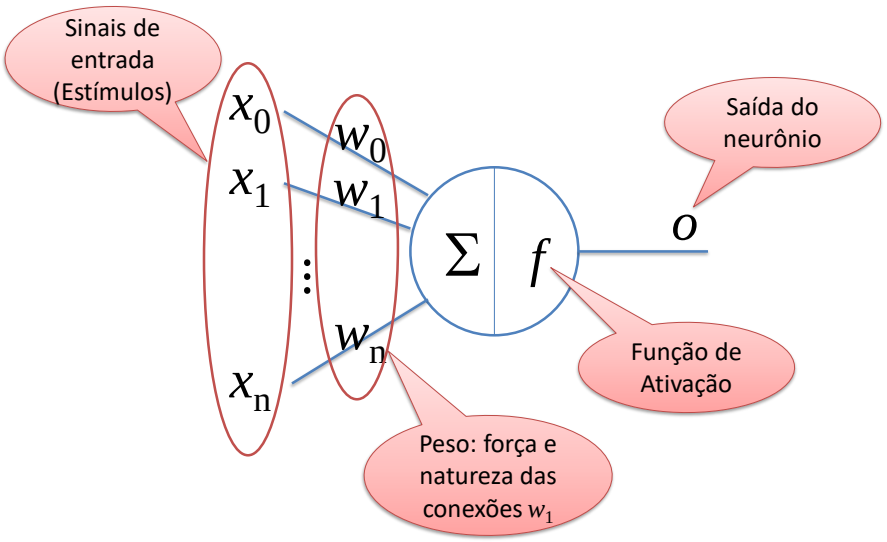


Comunicação Neural

- Potencial elétrico através da membrana da célula exibe picos.
- Pico se origina no corpo celular, passa pelo axônio, e faz com que os terminais sinápticos soltem neurotransmissores.
- Neurotransmissores passam através das sinapses para os dendritos de outros neurônios.
- Neurotransmissores podem ser excitadores ou inibidores.
- Se a entrada total de neurotransmissores para um neurônio é excitatória e ultrapassa um certo limite, ele dispara (tem um pico).

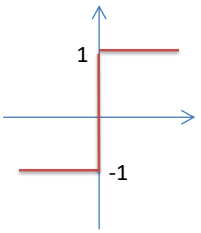


Neurônios "Artificiais"

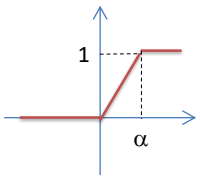


Funções de Ativação

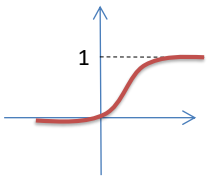
- Possíveis funções de ativação:



Hard Limiter
Degrau



Threshold Logic



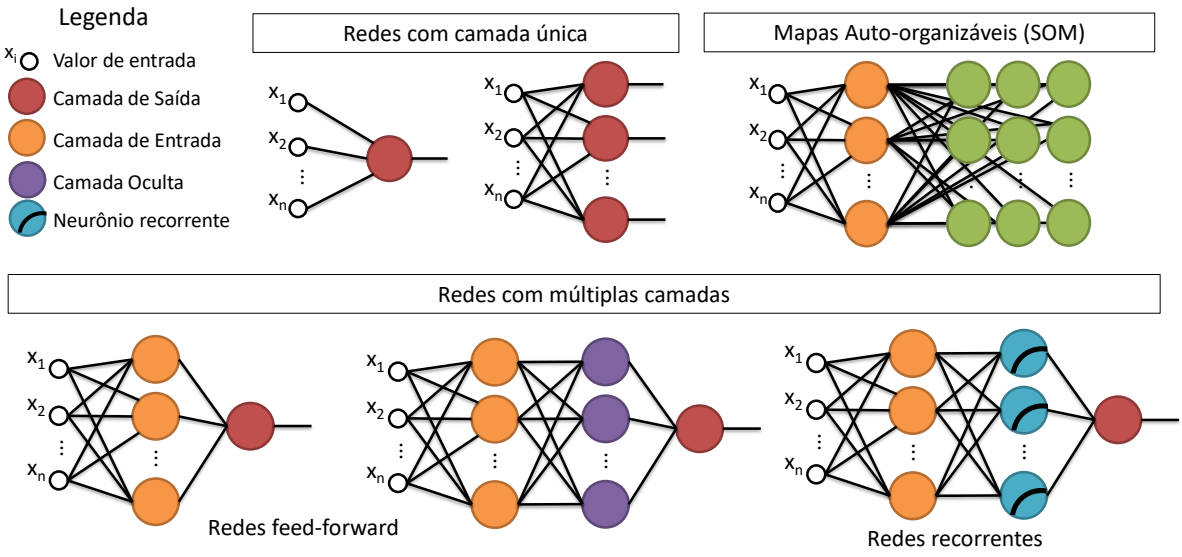
Sigmoid

Aprendizagem de Redes Neurais

- Aprendizagem Hebbiana: Quando dois neurônios conectados disparam ao mesmo tempo, a conexão sináptica entre eles aumenta.
  - *"Neurons that fire together, wire together."*
- Sinapses mudam de tamanho e força com experiência

11

Topologias das RNAs



PERCEPTRON

Computação Neural

- **McCollough** e **Pitts** (1943) mostraram como neurônios simples desse tipo (chamados *perceptrons*) poderiam calcular funções lógicas e serem usados como máquinas de estado.
  - Podem ser usados para simular portas lógicas:
    - AND: Todos w_{ji} são T_i/n, onde n é o número de portas.
    - OR: Todos w_{ji} são T_i
    - NOT: O limite é 0, entrada única com peso negativo
  - Podemos construir qualquer circuito lógico, máquina sequencial e computadores com essas portas.
    - Podemos representar qualquer função booleana usando uma rede com duas camadas de neurônios (AND-OR).

14

Aprendizagem de Perceptrons

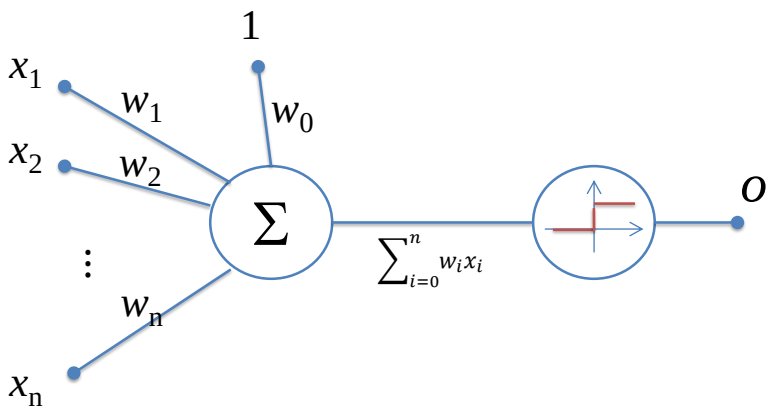
- Uma rede neural deve produzir, para cada conjunto de entradas apresentado, o conjunto de saídas desejado.
- O objetivo é aprender pesos sinápticos de tal forma que a unidade de saída produza a saída correta pra cada exemplo.
  - Quando a saída produzida é diferente da desejada, os pesos da rede são modificados

$$w_{t+1} = w_t + fator\_de\_correção$$

  - O algoritmo faz atualizações iterativamente até chegar aos pesos corretos.

15

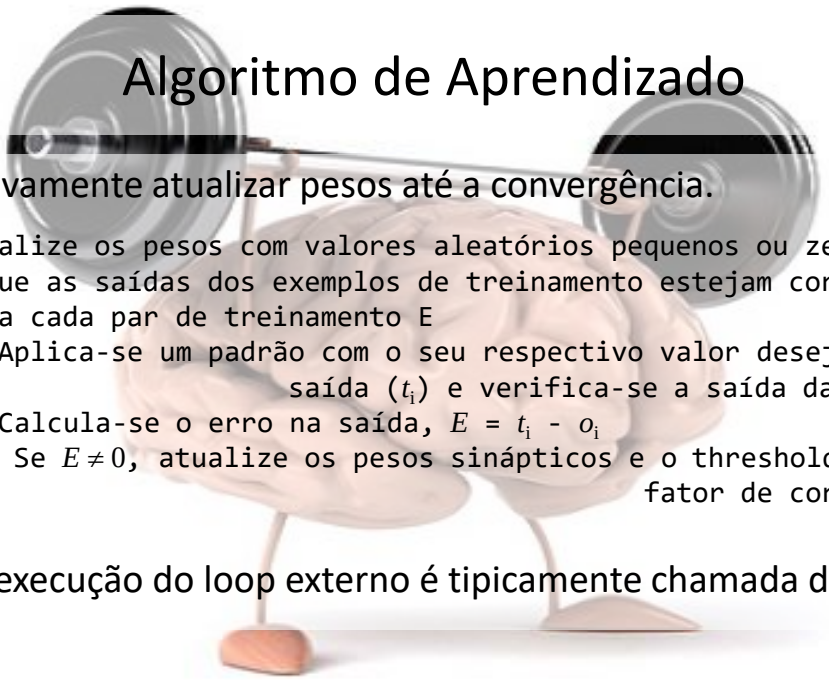
Modelo do Neurônio



Saída = 1, se $w_0 + w_1x_1 + ... + w_nx_n > 0$
Saída = 0, caso contrário

w_0 é o bias/threshold

Algoritmo de Aprendizado



- Iterativamente atualizar pesos até a convergência.

(1) Inicialize os pesos com valores aleatórios pequenos ou zero
(2) Até que as saídas dos exemplos de treinamento estejam corretos
(3) Para cada par de treinamento E
(4) Aplica-se um padrão com o seu respectivo valor desejado de saída (t_i) e verifica-se a saída da rede (o_i)
(5) Calcula-se o erro na saída, $E = t_i - o_i$
(6) Se $E \neq 0$, atualize os pesos sinápticos e o threshold com o fator de correção Δw_{ij}

- Cada execução do loop externo é tipicamente chamada de *época*.

17

Regra de Aprendizagem de Perceptrons

- Atualizar pesos usando:

$$w_{ij} = w_{ij} + \Delta w_{ij}$$

$$\Delta w_{ij} = \eta x_i (t_j - o_j)$$

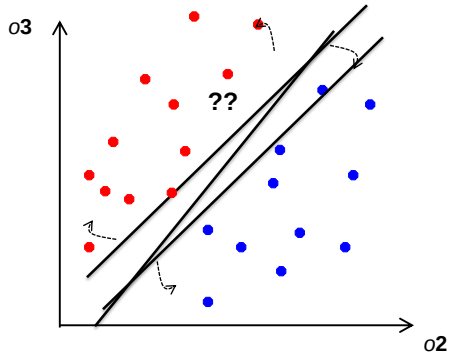
onde η é a "taxa de aprendizagem"
t_j é a saída especificada para a unidade j

- O processo equivale a:
  - Se a saída estiver correta, não fazer nada.
  - Se a saída estiver alta, baixar os pesos das saídas ativas
  - Se a saída estiver baixa, aumenta pesos das saídas ativas,

Erro: diferença entre o esperado e o obtido

Perceptron como Separador Linear

- Como o perceptron usa uma função de limite linear, ele procura por um separador linear que discrimine as classes.



$$w_{12}o_2 + w_{13}o_3 > T_1$$

$$o_3 > -\frac{w_{12}}{w_{13}}o_2 + \frac{T_1}{w_{13}}$$

ou *hiperplano* em um espaço *n*-dimensional

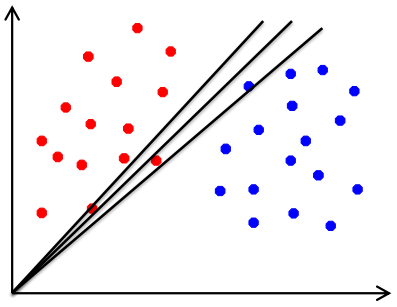
Para que serve o bias (w_0)?

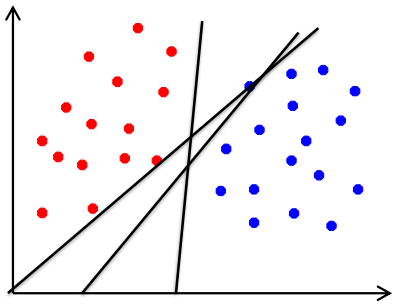
Sem bias

- Define um hiperplano passando pela origem

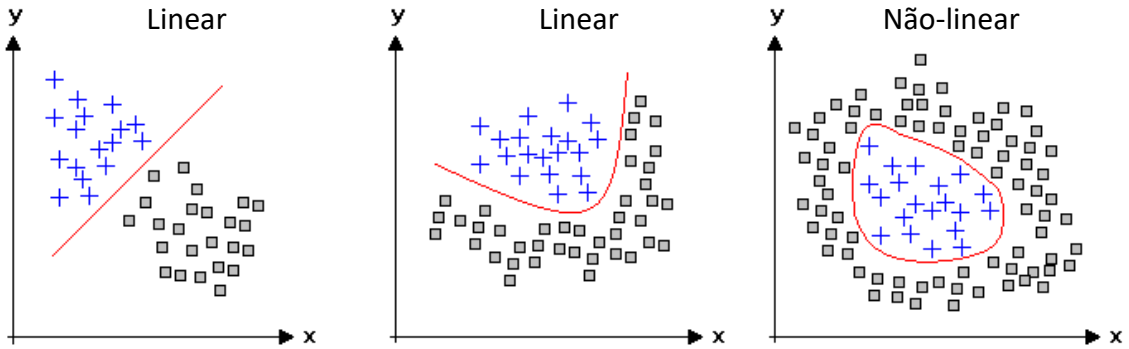
Com bias

- Permite que o hiperplano se desloque em relação a origem

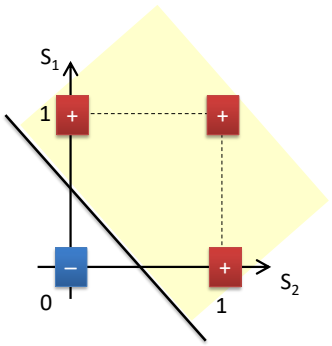




Problemas Lineares vs Não-lineares

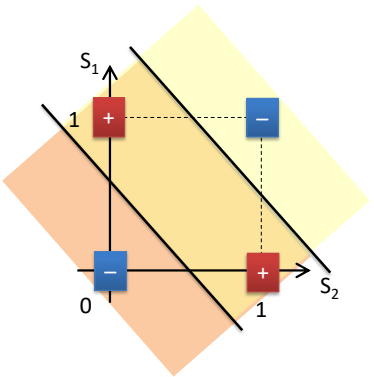


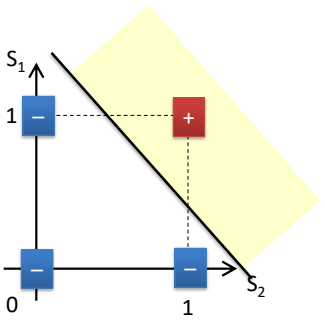
O problema do Ou-Exclusivo (XOR)



Operador OU

Operador XOR





Operador E

Limitações do *Perceptron*

- Obviamente não pode aprender conceitos que não é capaz de representar.
- **Minksy** e **Papert** (1969) escreveram um livro analisando o perceptron e descrevendo funções que ele não podia aprender.
- Esses resultados desencorajaram o estudo de redes neurais e as regras simbólicas se tornaram o principal paradigma de IA.
  - Tempos depois, descobriu-se que as redes de uma única camada funcionam para exemplos linearmente separados, mas redes multi-camadas podem representar qualquer função, mesmo não-lineares.

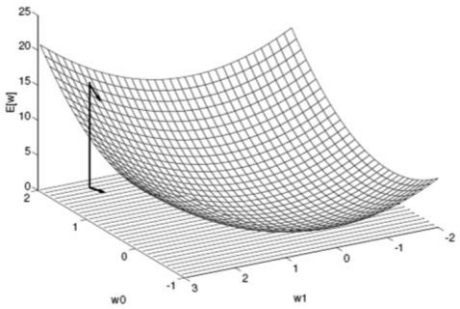
Teoremas

- **Teorema de convergência do perceptron**: Se os dados forem linearmente separáveis, então o algoritmo do perceptron irá corrigir para um conjunto consistente de pesos.

- **Teorema do ciclo do perceptron**: Se os dados não forem linearmente separáveis, o algoritmo irá repetir um conjunto de pesos e limites no final de uma época e, como consequência entra em um loop infinito.
  - Podemos garantir término do programa checando as repetições.

Perceptron como Subida de Encosta

- O espaço de hipóteses é um conjunto de pesos e um limite.
- O objetivo é minimizar o erro de classificação no conjunto de treinamento.
- O perceptron efetivamente realiza uma subida de encosta (descida) neste espaço.
- Para um único neurônio, o espaço é bem comportado com um único mínimo.



Desempenho do *Perceptron*

- Funções lineares são restritivas (bias ou viés alto) mas ainda razoavelmente expressivas; mais gerais que:
  - Conjuntiva pura
  - Disjuntiva pura
  - M-de-N (pelo menos M de um conjunto esperado de N características deve estar presente)
- Na prática, converge razoavelmente rápido para dados linearmente separáveis.
- Pode-se usar até resultados anteriores à convergência quando poucos *outliers* são classificados erroneamente.
- Experimentalmente, o Perceptron tem bons resultados para muitos conjuntos de dados.