Universidade Federal do Sul de Sudeste do Pará
Faculdade de Computação e Engenharia Elétrica
Inteligência Artificial

REDES NEURAIS ARTIFICIAIS (Parte 2)

PROF. DR. ELTON ALVES

Aprendizagem off-line e on-line

- **Off-line:**
  - **Os ajustes efetuados nos pesos e limiares dos neurônios só são efetivados após a apresentação de todo o conjunto de treinamento.**

- **On-line:**
  - **Os ajustes nos pesos e limiares dos neurônios são efetuados após a apresentação de cada amostra de treinamento.**
  - **É normalmente utilizada quando o comportamento do sistema a ser mapeado varia de forma bastante rápida.**

Perceptron

- **Um perceptron é um único neurônio que classifica um conjunto de entradas em uma de duas categorias (geralmente 1 ou -1).**

- **O perceptron geralmente usa uma função degrau, que retorna 1 se a soma ponderada das entradas exceder um limite e -1 caso contrário.**

$$X = \sum_{i=1}^{n} w_i \times x_i$$

$$Y = \begin{cases} +1 \; para \; X > t \\ 0 \; para \; X \leq t \end{cases}$$

Função descrita como degrau de X: $Degrau\ X = \begin{cases} +1 \; para \; X > t \\ 0 \; para \; X \leq t \end{cases}$

Perceptron

❑**O perceptron é treinado da seguinte forma:**

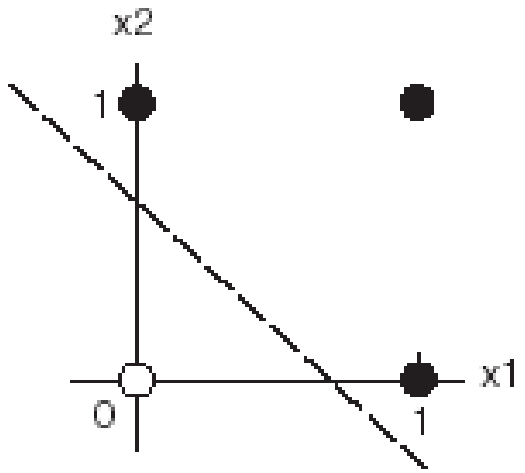
- **Primeiro, as entradas recebem pesos aleatórios (geralmente entre –0,5 e 0,5).**
- **Um item de dados de treinamento é apresentado. Se o perceptron classifica incorretamente, os pesos são modificados de acordo com o seguinte:**

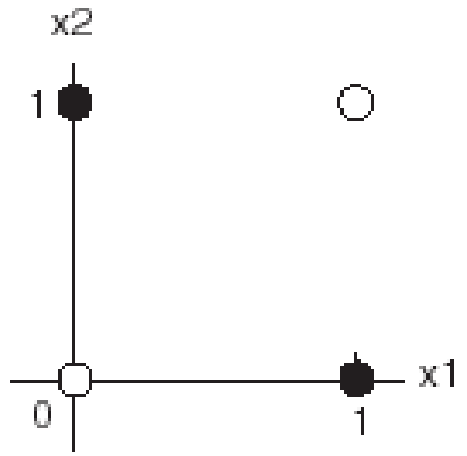
$$w_i \longrightarrow w_i + (a \times x_i \times e)$$

- **e é o tamanho do erro e a é a taxa de aprendizado, entre 0 e 1.**

Perceptron

- Perceptrons só podem classificar funções linearmente separáveis.
- O primeiro dos gráficos a seguir mostra uma função separável linearmente (OR).
- O segundo não é linearmente separável (Exclusivo-OU).





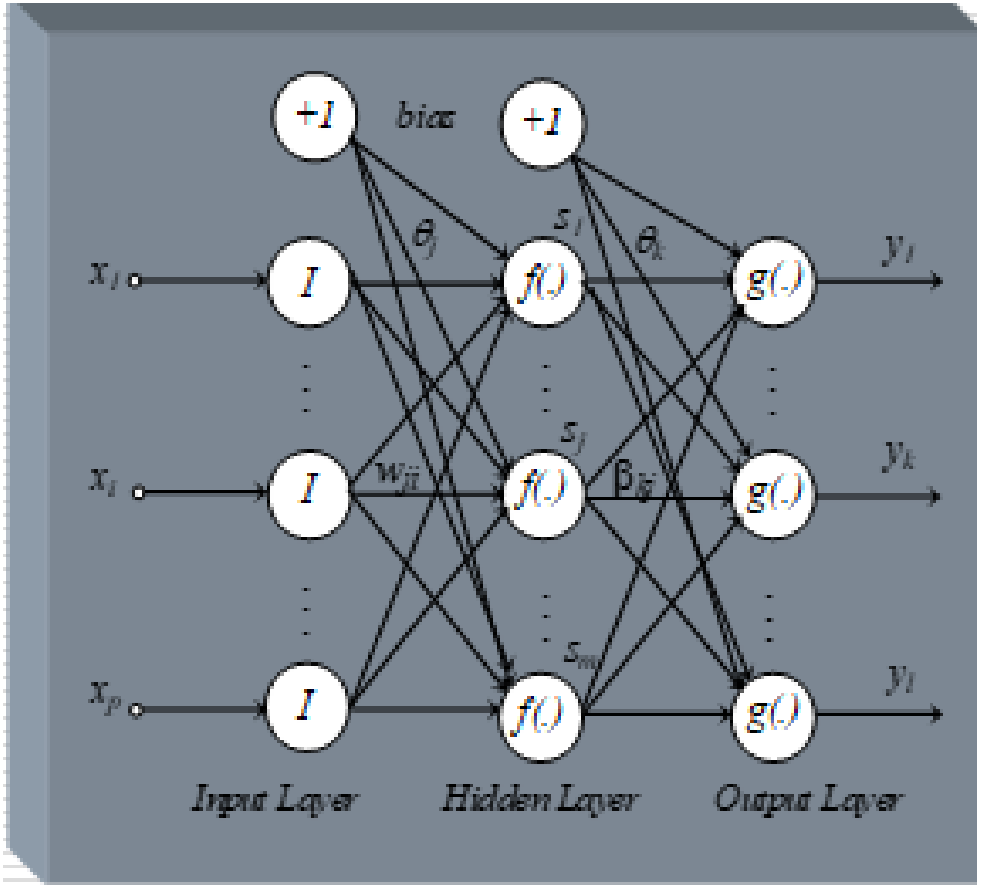
Perceptron Simples ao aprender a função OU

Época	X1	X2	Yesperado	Yatual	Erro	W1	W2
1	0	0	0	0	0	-0,2	0,4
1	0	1	1	1	0	-0,2	0,4
1	**1**	**0**	**1**	**0**	**1**	**0**	**0,4**
1	1	1	1	1	0	0	0,4
2	0	0	0	0	0	0	0,4
2	0	1	1	1	0	0	0,4
2	**1**	**0**	**1**	**0**	**1**	**0,2**	**0,4**
2	1	1	1	1	0	0,2	0,4
3	0	0	0	0	0	0,2	0,4
3	0	1	1	1	0	0,2	0,4
3	1	0	1	1	0	0,2	0,4
3	1	1	1	1	0	0,2	0,4

Perceptron de Múltiplas Camadas (MLP)

- Rede neural mais comumente utilizada;
- Considerada aproximadora universal;
- Treinada com algoritmo *Backpropagation*, que é considerado um marco no desenvolvimento das redes neurais;
- Algoritmo de aprendizado supervisionado;
- Regra de aprendizado por correção do erro.

MLP – Arquitetura Básica



Neurônio camada escondida:

$$s_j = f(\sum_{i=1}^{p} w_{ji} x_i + \theta_j)$$

Neurônio camada de saída:

$$y_k = g(\sum_{j=1}^{m} \beta_{kj} s_j + \theta_k)$$

MLP

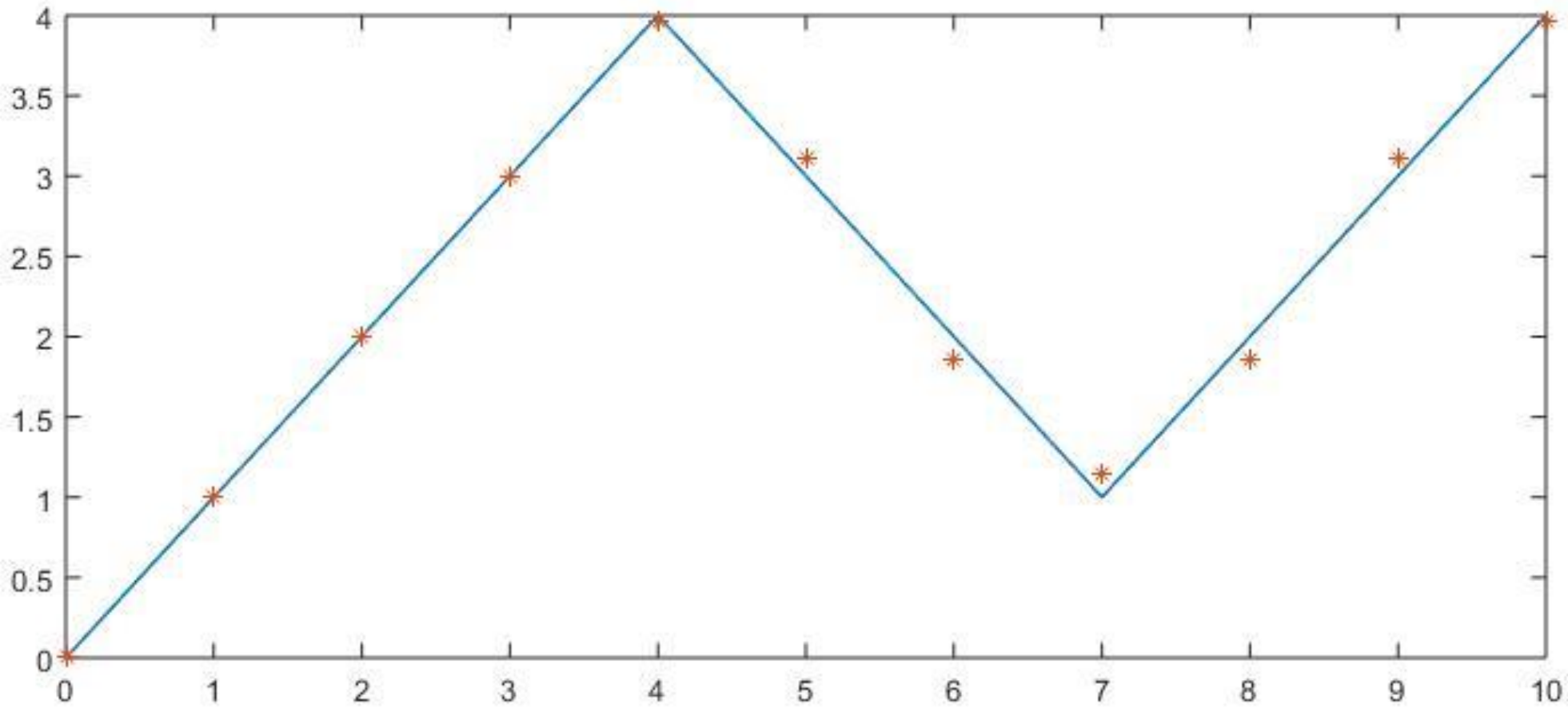
- Em uma MLP dois tipos de sinais podem ser identificados:

  - Sinal funcional: um sinal funcional é um sinal de entrada (estímulo) que chega na entrada e é propagado positivamente (neurônio a neurônio) através da rede, e aparece na saída como um sinal de saída.
  - Sinal de erro: os sinais de erro originam-se nas saídas e são retro-propagados (neurônio a neurônio) através da rede.

MLP – Capacidade de Aproximação Universal

- CYBENKO (1989) foi o primeiro pesquisador a demonstrar rigorosamente que uma rede MLP com uma única camada intermediária é suficiente para aproximar uniformemente qualquer função contínua.

MLP – Capacidade de Aproximação Universal



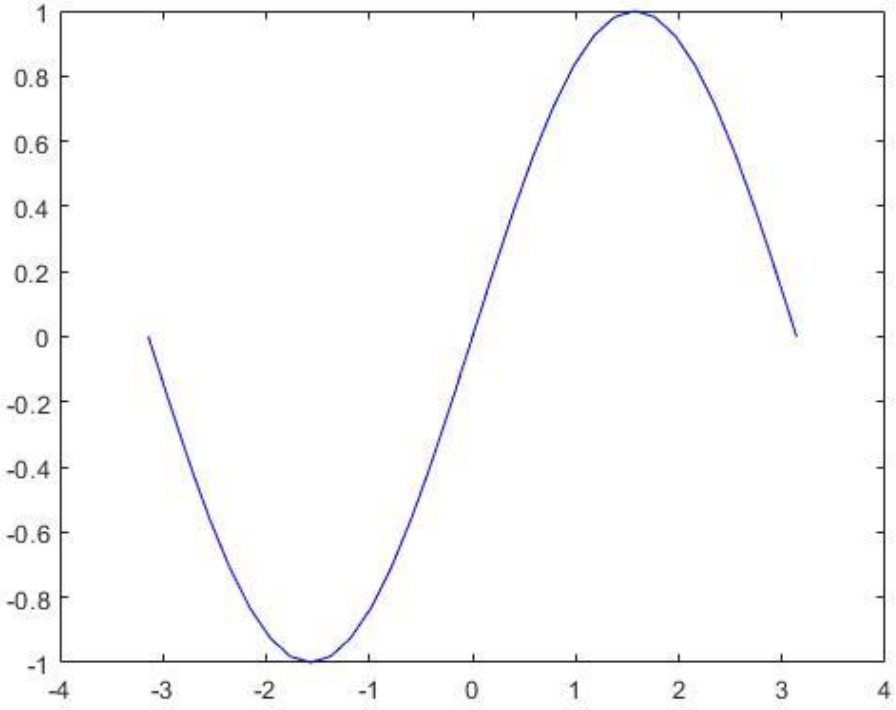
Algoritmo Backpropagation – Aprendizagem (retropropagação do erro)

- Algoritmo de aprendizagem composto principalmente de duas fases:

- **Fluxo da informação da entrada para saída da rede (para frente):** nesta fase a ativação dos neurônios escondidos são propagadas para os neurônios de saída.
  - Então o erro entre as saídas desejadas e a saídas da rede são calculados.
  - Pesos sinápticos são fixos.
- **Fluxo da informação da saída em direção dos neurônios escondidos (para trás):** nesta fase o erro é propagado para trás e os pesos conectando diferentes níveis são atualizados.

Atividade 3

1. **Explicar o princípio de funcionamento do algoritmo backpropagation (Formulação matemática), mostrando como este realiza o ajuste dos pesos da RNA-MLP.**
2. **Treinar uma RNA-MLP para a função** y = sin(x); x = -pi:pi/16:pi;



- **Data da entrega= 26/04/2022**

Taxa de aprendizado

- ***Quanto menor for o valor da taxa de aprendizado, menor serão as variações dos pesos sinápticos de uma iteração para outra, e mais suave será a trajetória no espaço de pesos – aprendizado lento***

- ***Quanto maior for o valor da taxa de aprendizado, maior serão as variações dos pesos sinápticos de uma iteração para outra, podendo tornar a rede instável***

Faixa da taxa de aprendizado : [0 1]

Alguns estudos e experiência de projetistas: taxa de aprendizado < 0.1

Critérios de Parada de Treinamento

Critérios de parada:

- ***Pelo número de ciclos de treinamento***

  treinamento termina após a execução de NI ciclos (épocas) de treinamento.

- ***Pelo valor do erro***

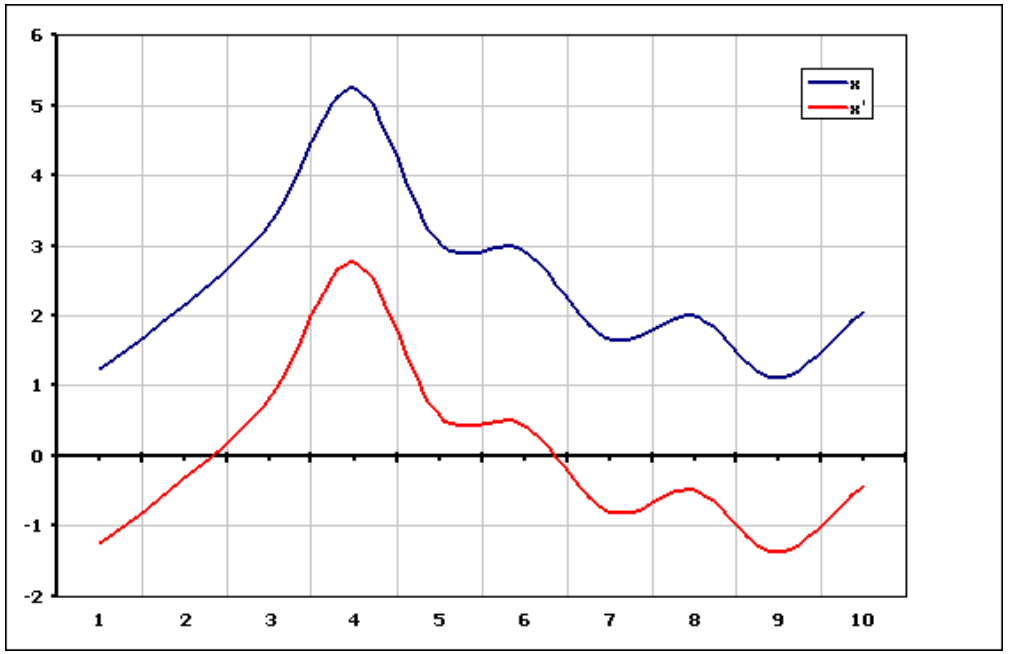
  o treinamento termina quando valor do erro é menor ou igual a α .

- ***Pelo critério de validação***

  Treinamento termina quando o erro no conjunto de validação para de decrescer

Normalização dos dados

1. Normalização com média zero

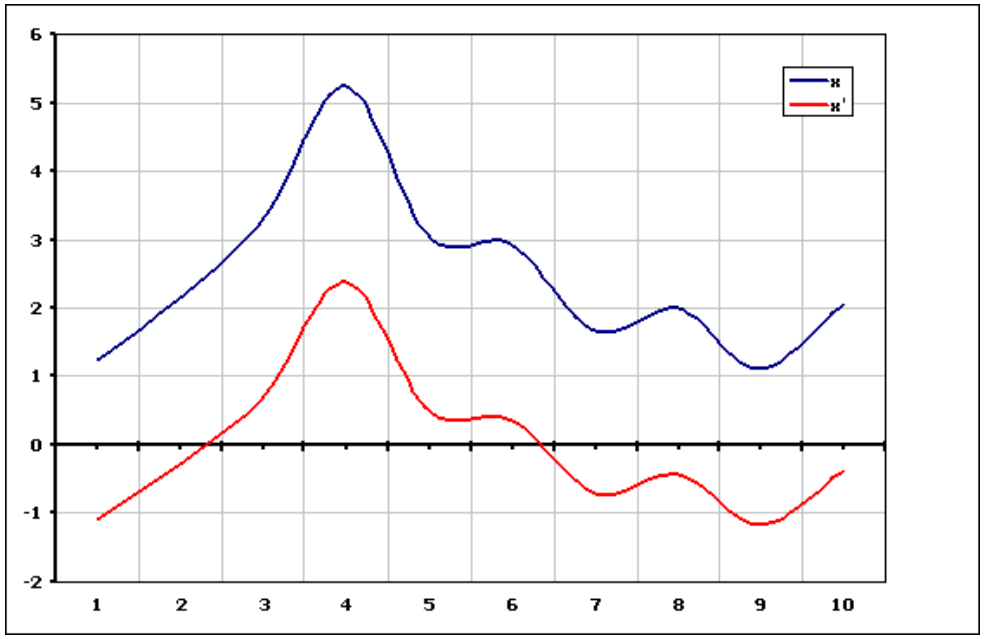


$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$x_i^{'} = x_i - u$$

Normalização dos dados

2. Normalização com média zero e Desvio padrão 1



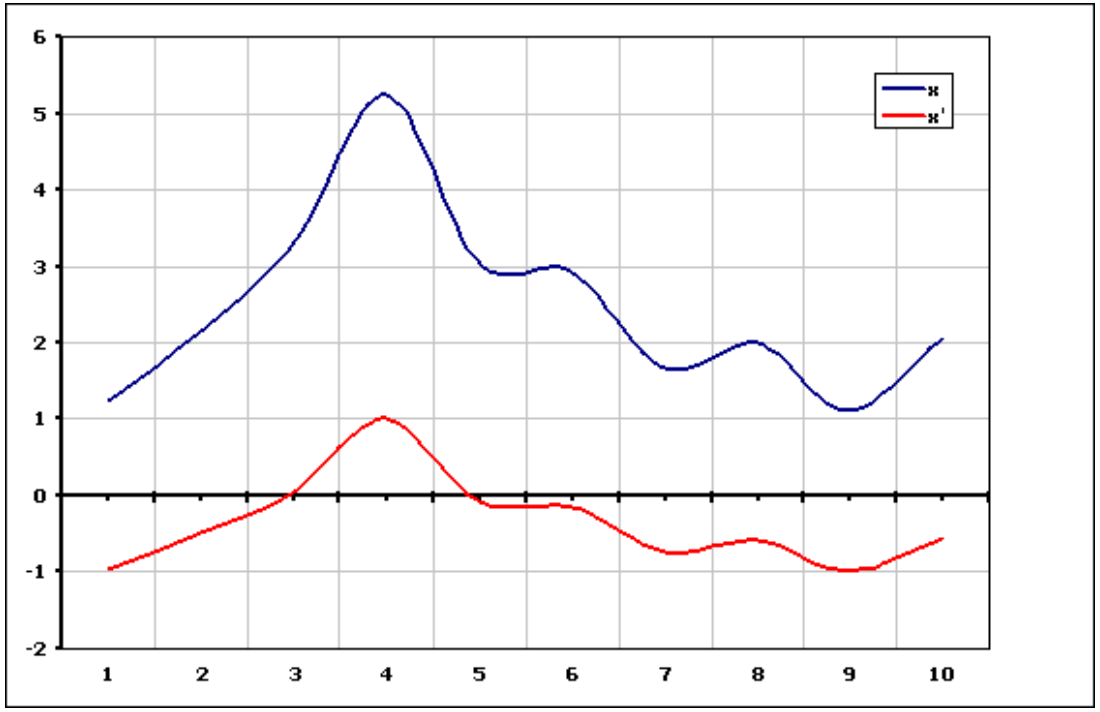
$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(x_i - u)^2}{N-1}}$$

$$x_i^{'} = \frac{x_i - u}{\sigma}$$

Normalização dos dados

3. Normalização dentro de um intervalo



$$L_{\min} \leq x_i^{'} \leq L_{\max}$$

$$x_i^{'} = \frac{(x_i - x_{\min})(L_{\max} - L_{\min})}{x_{\max} - x_{\min}} + L_{\min}$$

Generalização

- ***Habilidade para realizar predições para padrões de entrada que não fazem parte dos padrões utilizados para treinamento, mas que foram gerados da mesma distribuição entrada/saída que estes.***

- ***Considerada uma das maiores vantagens da rede neural.***

- ***Pobre generalização (sub-treinamento e sobre-treinamento)***

Generalização

Pobre generalização:

- ***Underfitting (sub-treinamento)*** ***complexidade da rede neural é inferior a complexidade do fenômeno que está sendo modelado.***

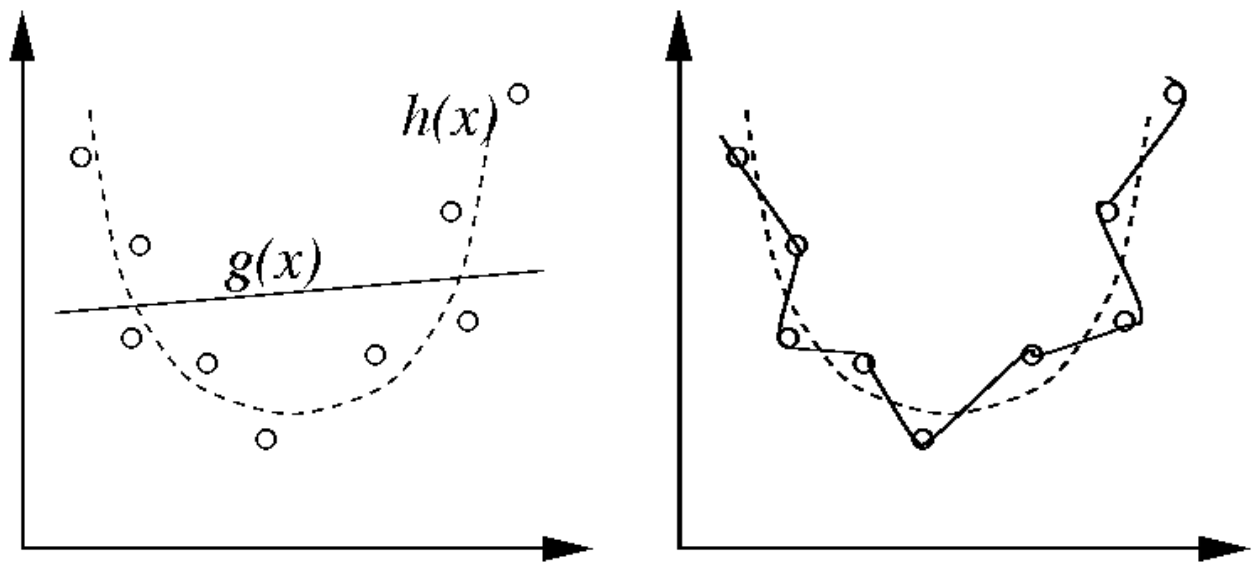
  O modelo tem um alto valor de bias

- ***Overfitting (sobre-treinamento)*** ***complexidade da rede neural é maior que a complexidade do fenômeno que está sendo modelado.***

  O modelo tem um valor alto de variância

Exemplo

- ***O conjunto de treinamento foi gerado a partir de uma função suave, h(x), com ruído adicionado***
- ***Queremos encontrar um modelo que aproxime h(x), dado um conjunto específico de dados y(x) gerado como:***
  - ***y(x) = h(x) + ε***



Técnica para evitar o overfitting

- ***Validação Cruzada*** *(Parada antecipada)*
  - *Dados divididos em três grupos: treino, validação e teste.*
  - *Grande número de dados disponível.*
  - *Conjunto de validação*

    Usado para testar a generalização da rede durante o treinamento.

    Se erro de validação aumenta, isto pode indicar sobre-treino e o treinamento deve ser então finalizado.