



# Implementação e análise de classificadores

Enrico Antonio J. M. de Moraes	9838219
José Luiz Carvalho de Sousa	9017273
Victor Angelo Nunes Notário	9836822



# Agenda

- 1) Introdução e objetivo
- 2) Preparo dos dados de entrada
- 3) Classificador Linear
- 4) Classificador Não-Linear

---

# Introdução & Objetivo

# Motivação



- Aprender a implementar o código de uma rede neural
- Tema muito discutido atualmente
- Métodos de minimização de uma função custo, conforme discutido na disciplina

# Objetivo



- Apresentar os resultados de dois classificadores:

**Não-Linear:** utiliza-se o algoritmo BackPropagation

**Linear:** classificador multiclasse do MATLAB

- Comparação de resultados
- Dificultar o aprendizado com mais entradas em paralelo!

# Referências



- Artigo “Deep learning” por Yann LeCun, Yoshua Bengio & Geoffrey Hinton.
- [https://www.youtube.com/watch?v=aircAruvnKk&ab\\_channel=3Blue1Brown](https://www.youtube.com/watch?v=aircAruvnKk&ab_channel=3Blue1Brown)

---

# Preparo dos dados de entrada

# Entradas em paralelo para treino: 4 ou 9 entradas

treina.txt - Bloco de Notas

Arquivo	Editar	Formatar	Exibir	Ajuda
6.3	2.8	5.1	1.5	3
6.1	2.6	5.6	1.4	3
5.4	3.4	1.7	0.2	1
5.6	2.9	3.6	1.3	2
5.8	2.7	4.1	1.0	2
5.1	3.7	1.5	0.4	1
5.0	3.0	1.6	0.2	1
6.3	2.7	4.9	1.8	3
6.7	3.3	5.7	2.1	3
5.1	3.8	1.5	0.3	1
7.2	3.2	6.0	1.8	3
6.2	2.8	4.8	1.8	3

teste.txt - Bloco de Notas

Arquivo	Editar	Formatar	Exibir	Ajuda
4.6	3.4	1.4	0.3	
5.5	2.5	4.0	1.3	
5.5	2.6	4.4	1.2	
5.0	3.4	1.5	0.2	
4.4	2.9	1.4	0.2	
5.7	2.8	4.1	1.3	
7.6	3.0	6.6	2.1	
4.9	2.5	4.5	1.7	
7.3	2.9	6.3	1.8	
4.8	3.0	1.4	0.1	
4.5	2.3	1.3	0.3	
6.3	2.3	4.4	1.3	



Alvos da classificação!

---

# Classificador Não Linear



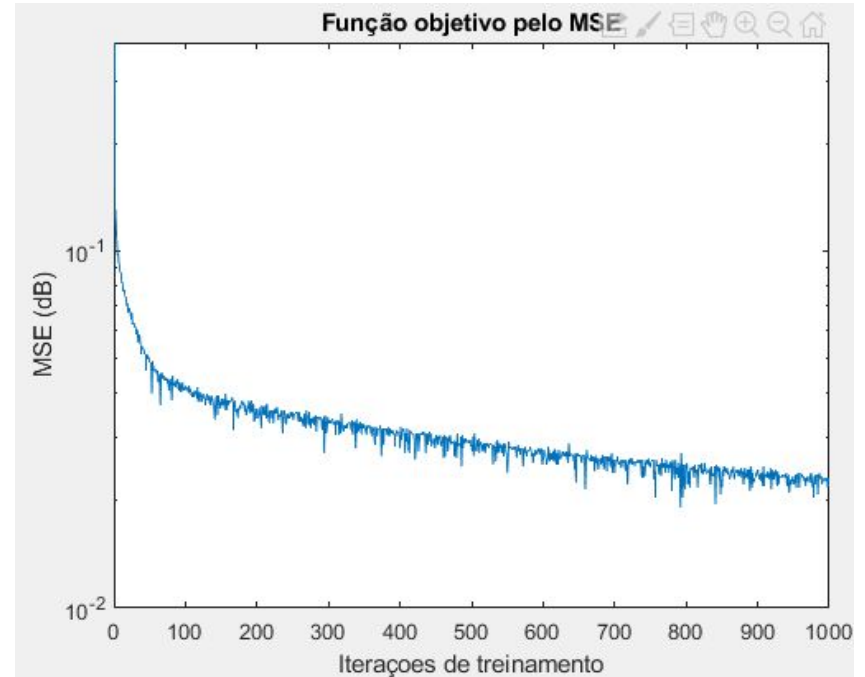
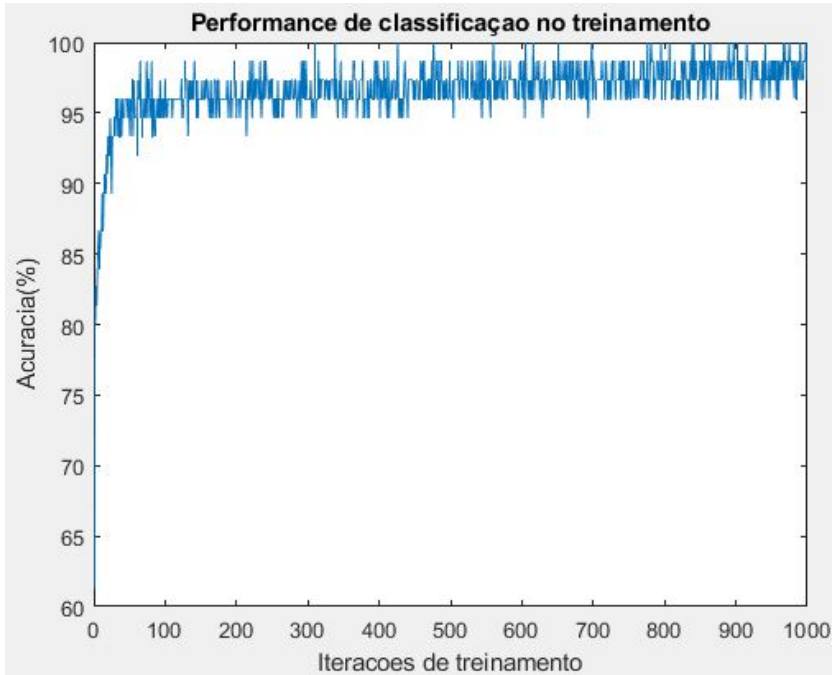
# Classificador Não Linear



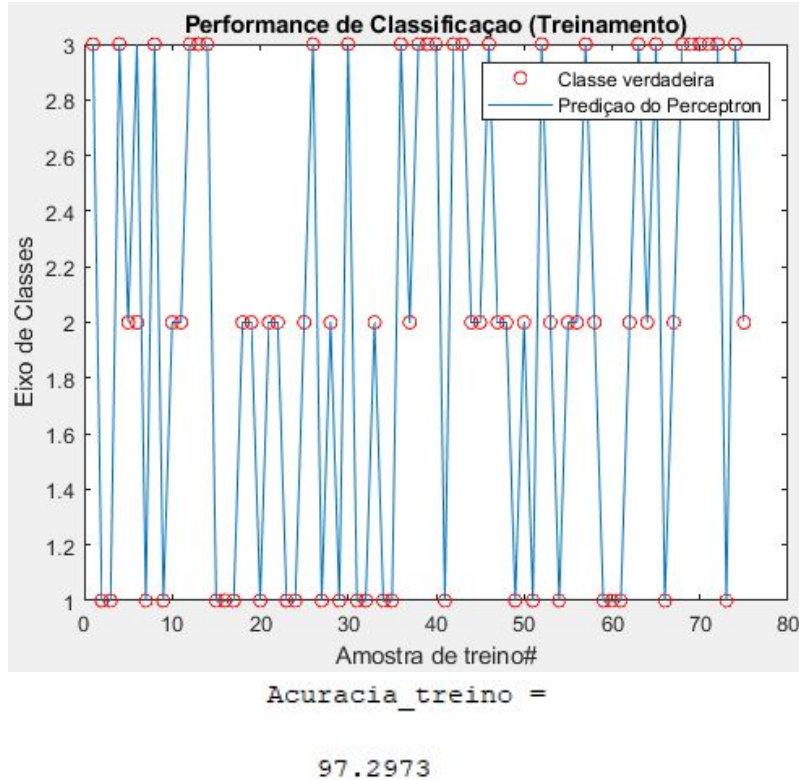
- Uma camada de entrada com 4 ou 9 neurônios, uma camada '*hidden*', um neurônio de saída
- Função de ativação: Tangente hiperbólica (3 classes: -1, 0 e 1)  
Em cada iteração, fazer:
  - 1º - Aplicação da função de ativação nas camadas de entrada e intermediária
  - 2º - Aplicação da derivada da função de ativação, primeiro na camada de saída.
  - 3º - '*Retropropagar*' a derivada para as camadas anteriores, associando o peso da camada anterior com a atual.
  - 4º - Atualizar os pesos das camadas de entrada e *hidden*
  - 5º - Calcular o erro quadrático médio
- Código extenso, transcrito no relatório

# Classificador Não Linear

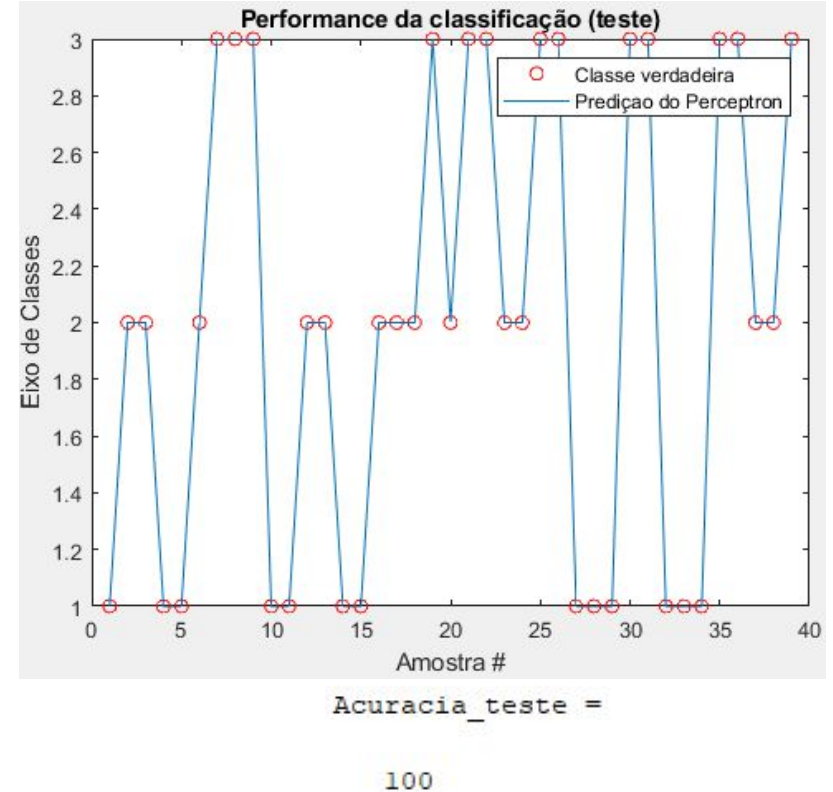
Para 4 entradas em paralelo:



# Classificador Não Linear

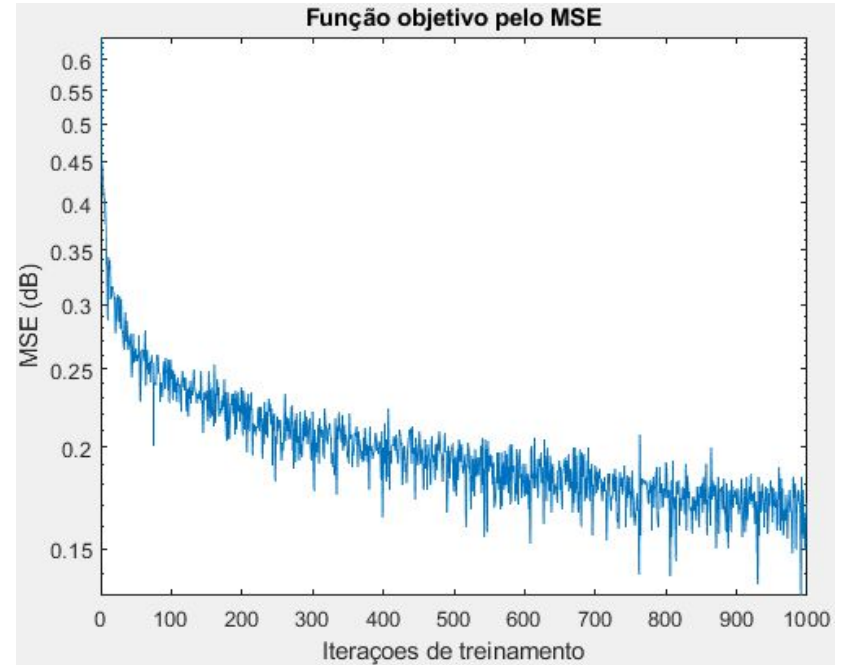
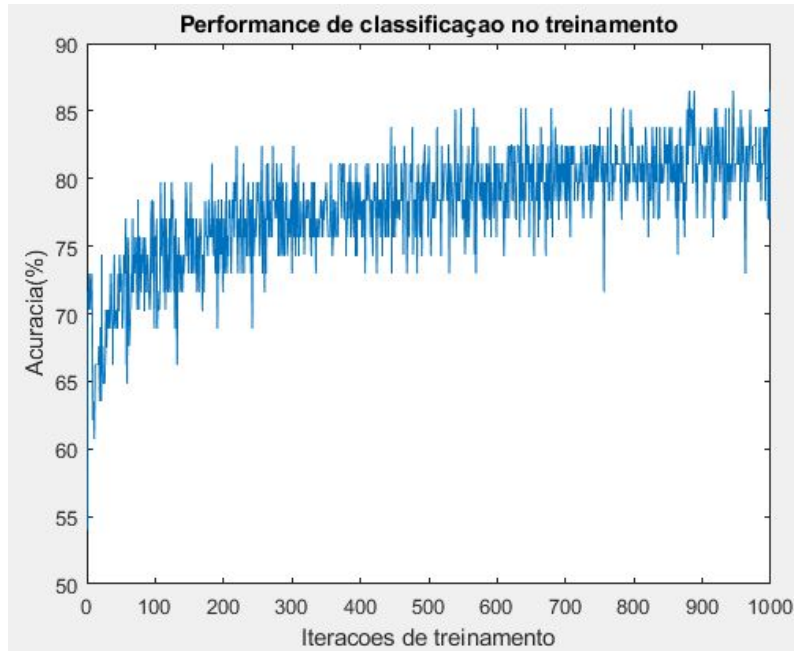


Para 4 entradas em paralelo:



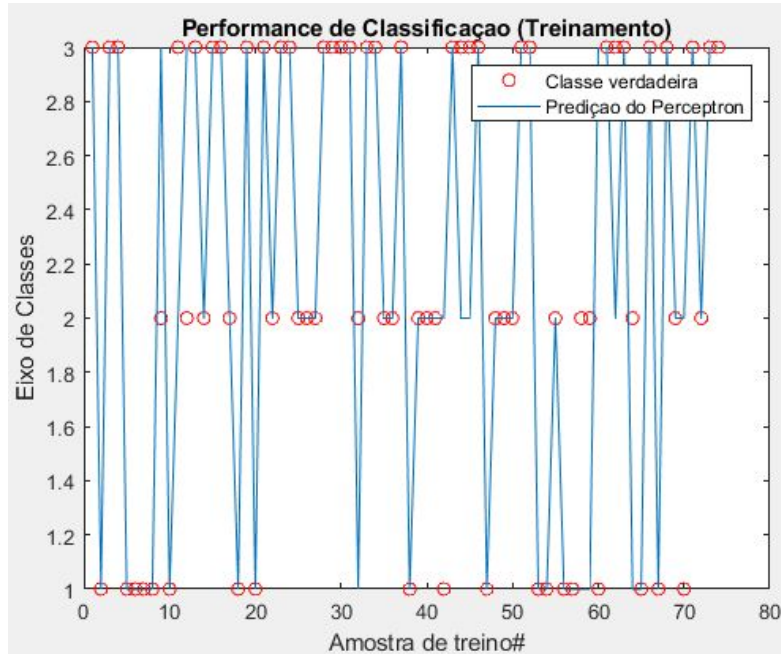
# Classificador Não Linear

Para 9 entradas em paralelo:



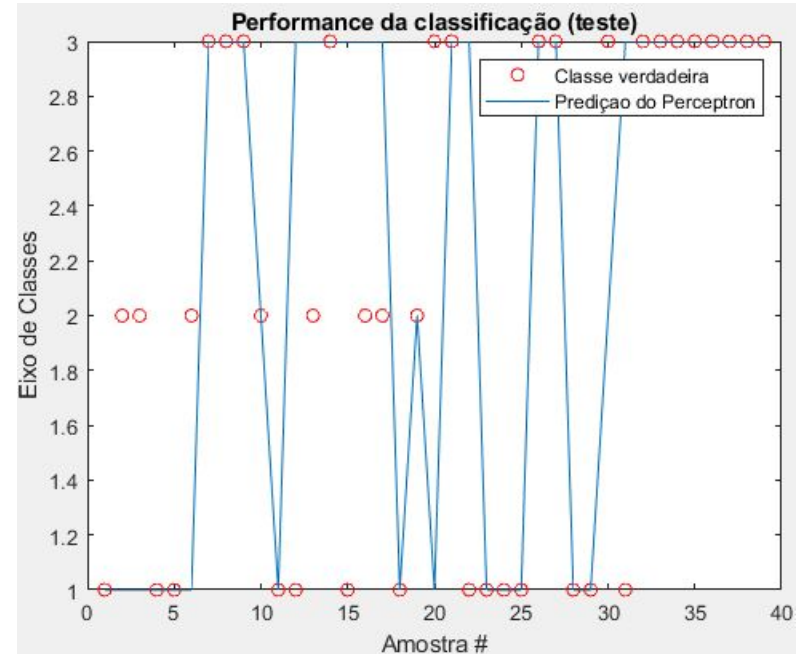
# Classificador Não Linear

Para 9 entradas em paralelo:



Acuracia\_treino =

82.4324



Acuracia\_teste =

69.2308

---

# Classificador Linear

# Classificador Linear

- Utilização do comando *fitcecoc* do MATLAB
- Treinamento e teste direto da rede
- Inserção dos dados preparados para o aprendizado supervisionado

```
%%% Classificador linear - PTC3101 - Professor Felipe Pait

clc
clear all
close all

%Data_treino = load('treinal.txt') %Utilizado para 9 entradas
%Data_teste = load('teste1.txt')
Data_treino = load ('treina.txt');
Data_teste = load ('teste.txt');

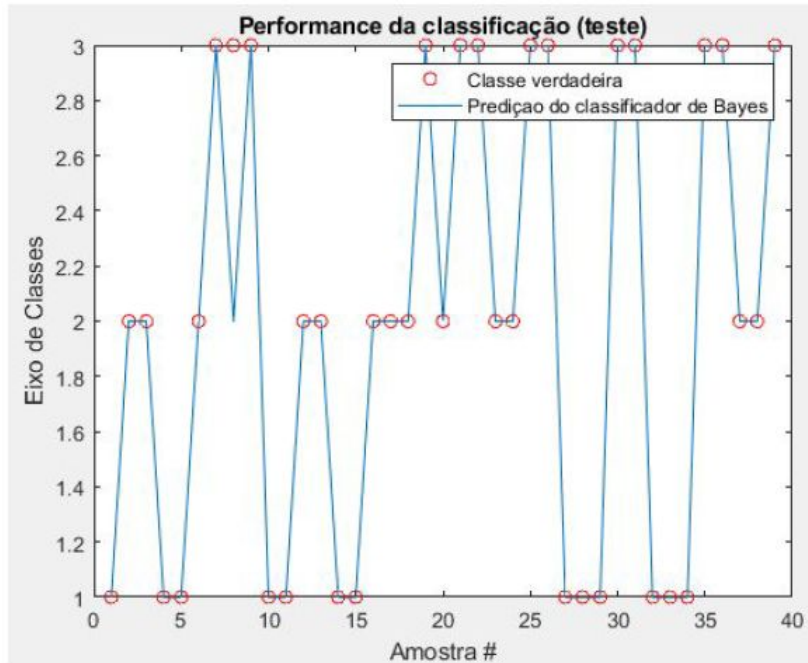
Dados_treino = Data_treino(:,1:4); %leitura dos dados de treinamento
Alvos_treino = Data_treino(:,5)';
Dados_teste = Data_teste(:,1:4); %leitura dos dados de teste e
exclusão da linha 5 no arquivo de teste
Alvos_teste = Data_teste(:,5)'; %classes verdadeiras do teste
Dados = [Dados_treino Alvos_treino'];

Classificador_Linear_treinado = fitcecoc(Dados_treino, Alvos_treino);
%utilização da função do deep Learning Toolbox
Classes_obtidas_teste =
predict(Classificador_Linear_treinado, Dados_teste); %predição dos
dados de teste pelo classificador treinado

Classificador_Linear_treinado = Classes_obtidas_teste';
Erros = [Alvos_teste - Classes_obtidas_teste'] % Vetor de Erros

%% plotagem das amostras e classes obtidas %%
figure
plot(Alvos_teste, 'or')
hold on
plot(round(Classes_obtidas_teste))
legend('Classe verdadeira', 'Predição do classificador linear')
xlabel('Amostra #')
ylabel('Eixo de Classes')
title('Performance da classificação (teste)')
Acuracia_teste =
length(find((round(Classes_obtidas_teste') - Alvos_teste) == 0)) * 100 / length(Classes_obtidas_teste);
Acuracia_teste
```

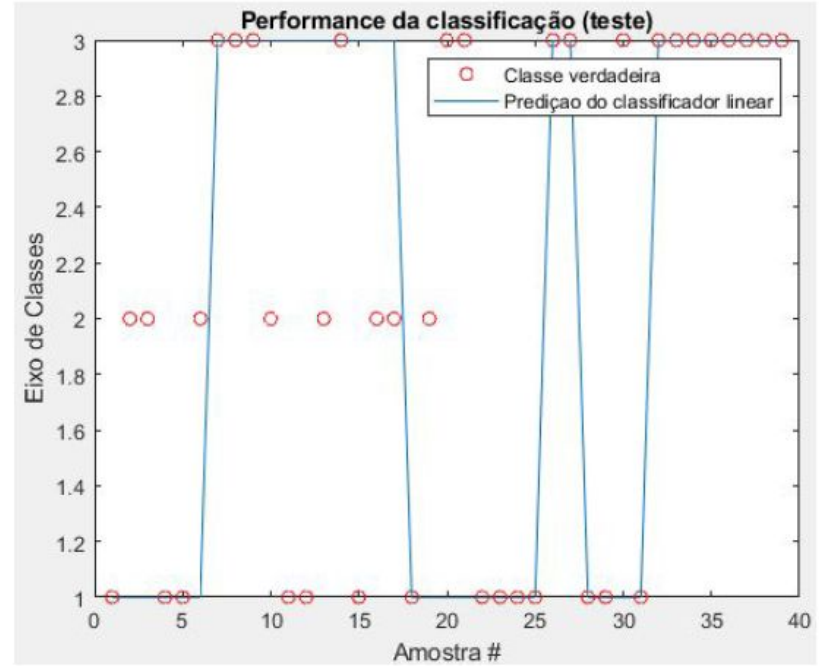
## Para 4 entradas:



Acuracia\_teste =

97.4359

## Para 9 entradas:



Acuracia\_teste =

64.1026



---

# Conclusões

# Conclusões



- O classificador linear apresentou bom desempenho.
- A classificação por meio do BackPropagation apresentou-se mais robusta e precisa.
- Para 4 entradas balanceadas em paralelo, ambos os algoritmos cumprem adequadamente seus papéis de classificação
- Quando inserimos 9 entradas não balanceadas em paralelo, os dois algoritmos acabam se perdendo no treinamento, o que prejudica a classificação final
- A amostra de treinamento interfere diretamente no desempenho do classificador!