Projeto de Redes Neurais e MATLAB

Centro de Informática
Universidade Federal de Pernambuco
Sistemas Inteligentes – IF684

Arley Ristar – arrr2@cin.ufpe.br
Thiago Miotto – tma@cin.ufpe.br
Baseado na apresentação de Alice Lucena

Objetivos

Projeto da disciplina

- O que é o MATLAB?
- Como usar o projeto? Aliás, o que será feito?

Objetivos

Projeto da disciplina

- Valor: 1 ponto
  - 50% relatório (coletivo)
  - 50% perguntas aleatórias (individual)

MATLAB

- MATrix LABoratory é um software de alta performance voltado para o cálculo numérico.
- Ele integra análise numérica, cálculo com matrizes, processamento de sinais e construção de gráficos.
- Problemas e soluções são expressos matematicamente através de matrizes.

MATLAB

- É uma linguagem interpretada, ou seja, cada comando é lido e interpretado um por vez.
- Comandos são escritos na janela de comando.
- Tudo é considerado matriz. Dados escalares são considerados com matrizes 1x1.

  Ex: x = 10; x = [1 2 3]; x = ‘final’

Arquivos *.m

- Os comandos do matlab são normalmente digitados na janela de comando.
- Apenas uma linha de comando é introduzida na janela que posteriormente é interpretada.
- Porém, o matlab oferece a opção de executar seqüências de comandos armazenadas em arquivos.

Arquivos *.m

- Os arquivos que contêm essas declarações são chamados de arquivos “.m” ou também scripts.
- Eles consistem de uma seqüência de comandos normais do matlab.

Exemplo:

O script que será usado por vocês para treinarem a rede neural.

Gráficos

- O matlab oferece a opção para visualizar gráficos.
- Há uma lista com vários comandos para plotar diferentes tipos de gráficos.
- Todos esses comandos recebem como argumento um vetor numérico.

Projeto da Disciplina

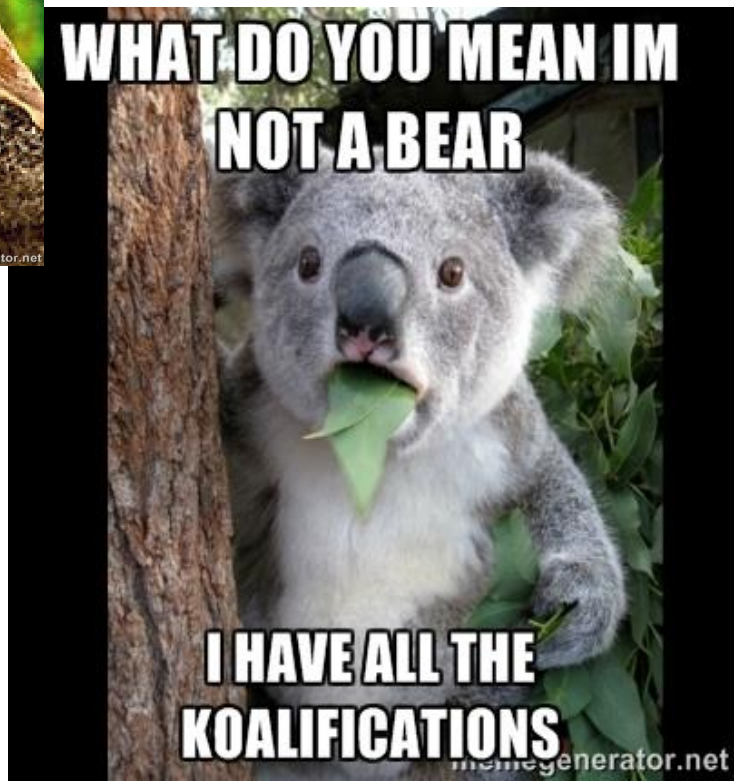
- Serão usados os problemas disponíveis na conhecida base de dados *Proben1.*
- Cada problema possui 3 arquivos de dados.
  Ex: O problema câncer possui os arquivos cancer1.dt, cancer2.dt e cancer3.dt.
- Os arquivos diferem na ordem de apresentação dos padrões, dependendo desta ordem a rede neural pode gerar resultados diferentes.
- **Não haverá** equipes com o mesmo arquivo de dados. As equipes terão no máximo 5 integrantes e a equipe deverá mandar email para Arley (arrr2@cin.ufpe.br) com o subject "[SI] Equipe" informando quais são os alunos que compõem a equipe. Assim que o email for recebido**, será enviado ao grupo o nome do arquivo referente ao seu projeto**.

O que é isso?

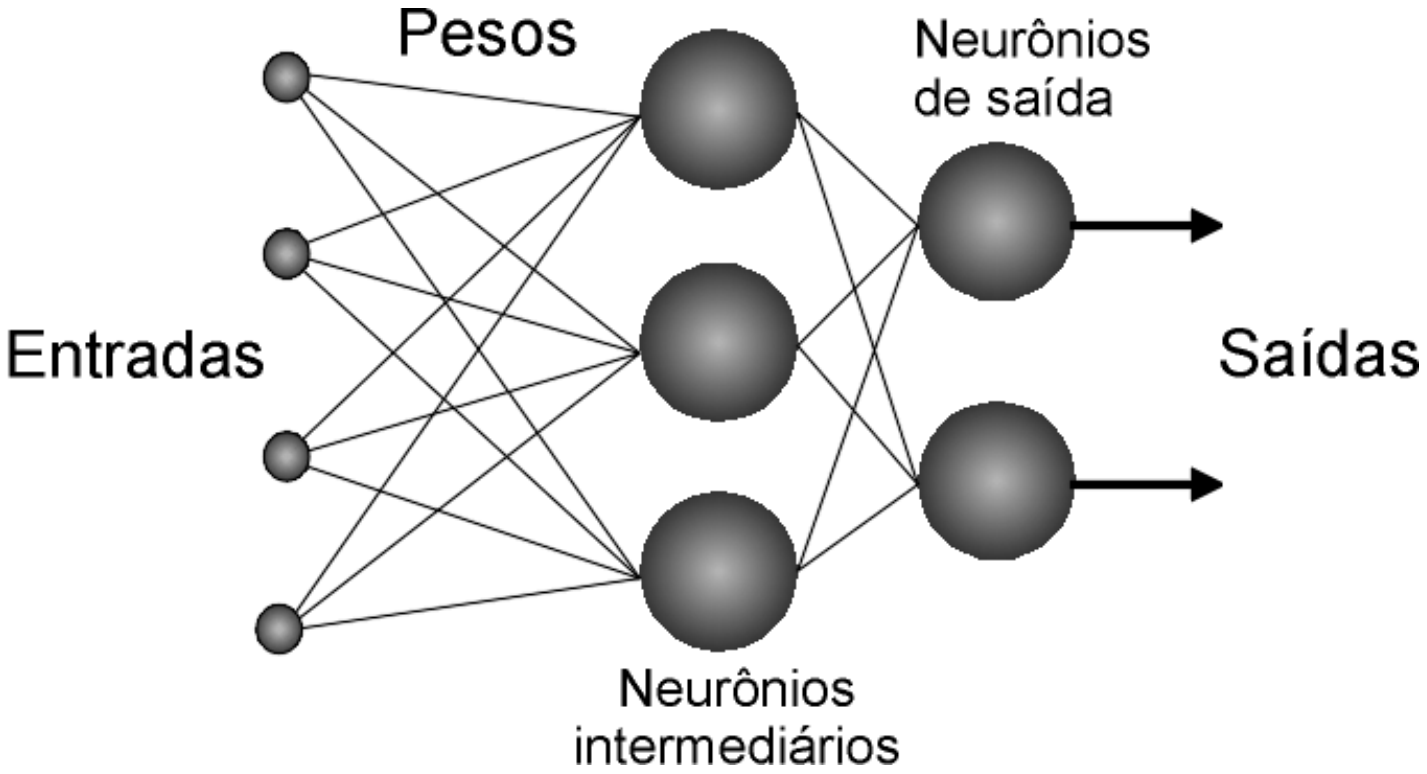
E isso?

Como vocês sabem?

Como se sabe que um urso é um urso?



Entendi nada...



Peso	Tamanho	...	Agressivo	Classe
500kg	2m	...	Sim	É urso
550kg	2m10cm		Não	Não é urso
⋮	⋮	⋮	⋮	⋮
600kg	1m80cm	...	Não	É urso

Pré-processamento

Neste projeto, não será cobrado, pois os dados já foram pré-processados.

É comum fazer normalização (para garantir que os valores estarão dentro de uma determinado intervalo). Nos problemas do *Proben* é usado o método min-max[0,1].

Exemplo de escalonamento para o intervalo [0,1]:

$$x_{norm} = \frac{(x - x_{\min})}{(x_{\max} - x_{\min})}$$

Onde x_{norm} é o valor normalizado correspondente ao valor original x, e x_{min} e x_{max} são os valores mínimo e máximo entre todos os valores (ou separadamente por atributo).

Particionamento dos Dados

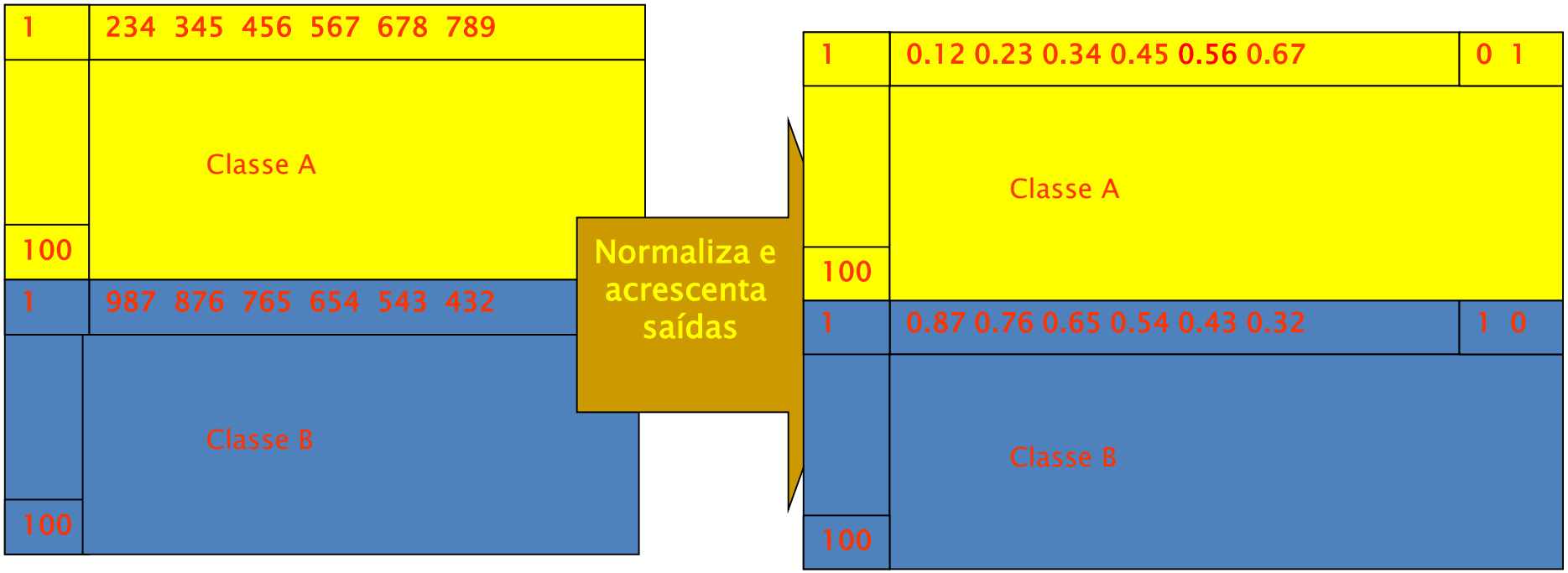
- **Particionamento de dados utilizado no *Proben1*:**
  - 50% dos padrões de cada classe escolhidos aleatoriamente para treinamento,
  - 25% para validação,
  - 25% para teste.

  É importante que as proporções entre as classes no conjunto completo de dados sejam mantidas nos conjuntos de treinamento, validação e teste.

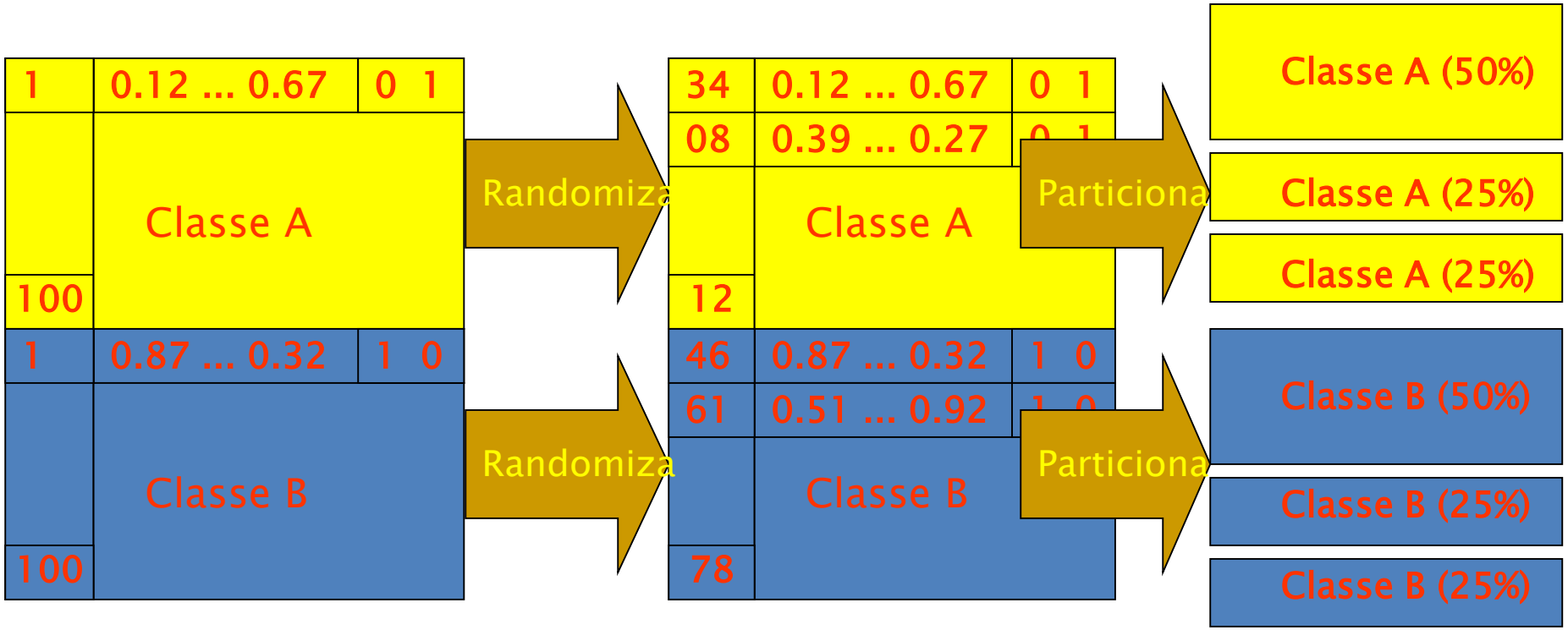
Neste projeto, não será cobrado, pois cada arquivo de dados já está dividido em treinamento, validação e teste.

Normalização dos Dados

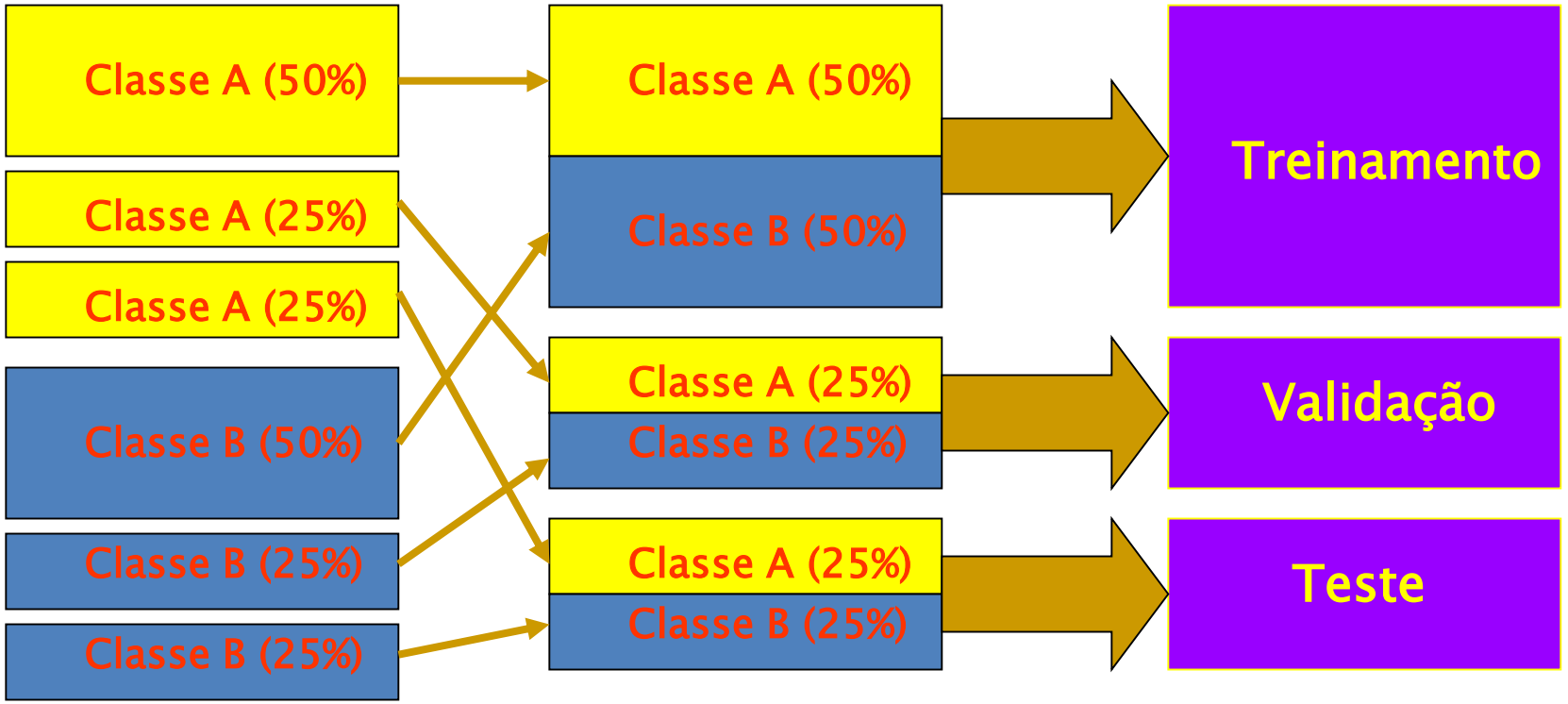
Exemplo:



Randomiza e Particiona



Divisão dos Dados



Definição da Topologia MLP

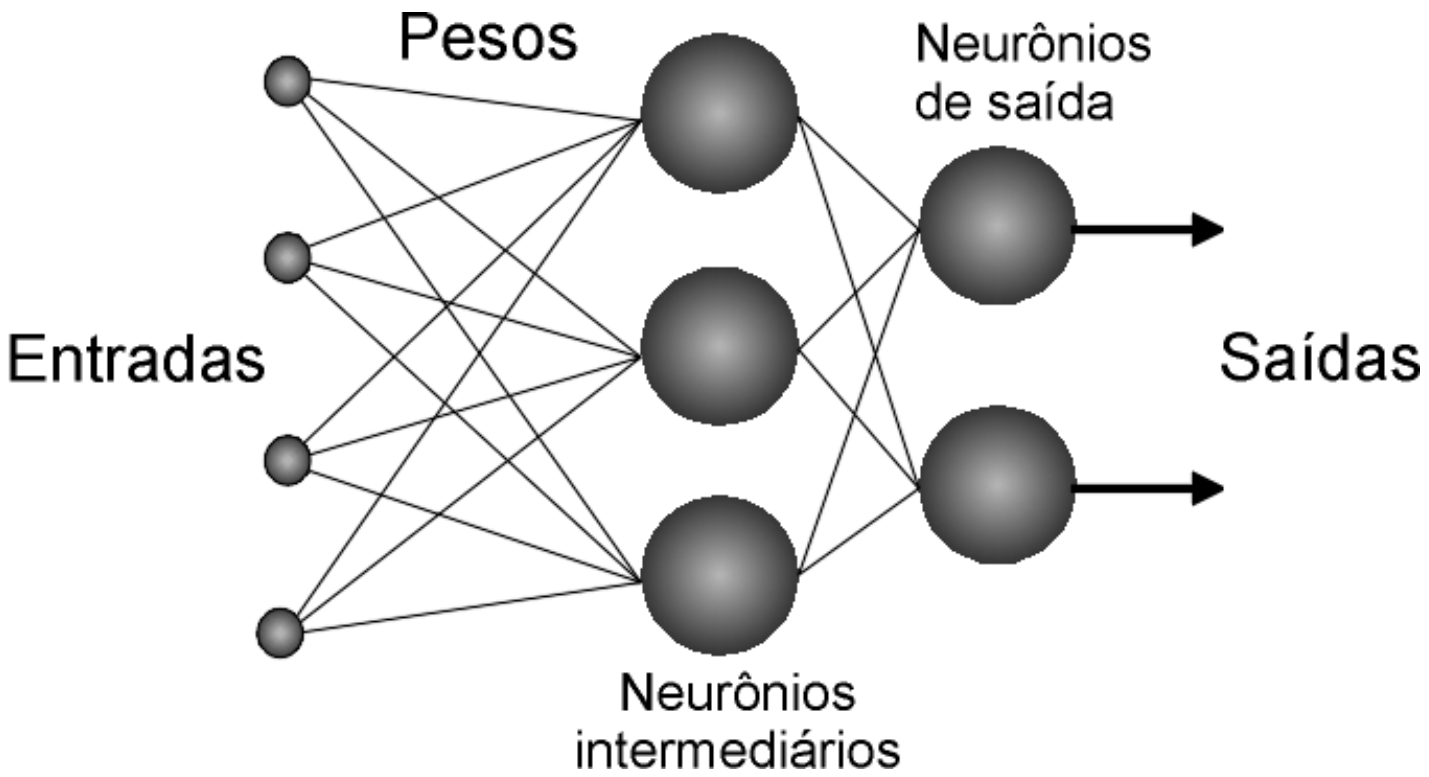
Aspectos que serão fixos neste projeto:

- **Nº de nodos de entrada:** Quantidade de atributos de entrada.
- **Nº de nodos de saída:**
  - Em problemas de classificação, é a quantidade de classes.
    - Regra de classificação *winner-takes-all*: o nodo de saída que gerar a maior saída define a classe do padrão.
  - Em problemas de aproximação, é a quantidade de variáveis de saída.
- **Uma única camada escondida**
- **Função de ativação dos neurônios: sigmóide logística.**
- **Todas as possíveis conexões entre camadas adjacentes, sem conexões entre camadas não-adjacentes.**

Começando a entender?

Em que não iremos mexer?

Nº de nodos de entrada



Variação

Aspectos que serão variados neste projeto:

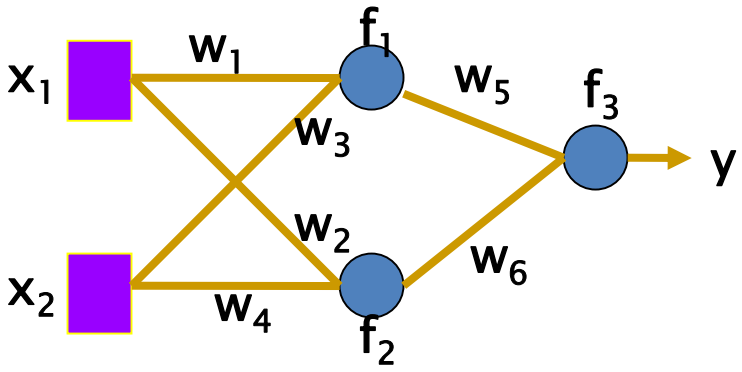
- **Nº de neurônios escondidos (serão usados 3 valores);**
- **A taxa de aprendizado a ser utilizada (serão usados 3 valores);**
- **O número de máximo de iterações.**

Número de neurônios na camada escondida

Variando o nº de neurônios escondidos, estamos variando a quantidade de pesos da rede.

Explicação: Uma rede neural implementa uma função.

- Os pesos da rede são os parâmetros da função.
- Dessa forma, aumentar a quantidade de pesos da rede significa aumentar a complexidade da função implementada.



As funções f_i são do tipo sigmóide logística.

$$y = f_3(w_5 f_1 (w_1 x_1 + w_3 x_2) + w_6 f_2 (w_2 x_1 + w_4 x_2)).$$

Underfitting e Overfitting

ATENÇÃO!

- **Se a quantidade de pesos for pequena demais, pode haver *underfitting*.**
  - A função implementada não tem complexidade suficiente para resolver o problema abordado.
- **Se a quantidade de pesos for grande demais, pode haver *overfitting*.**
  - A função implementada tem complexidade demais para o problema, sendo capaz de modelar detalhes demais dos dados de treinamento.

A taxa de aprendizado a ser utilizada

Usando taxa de aprendizado muito baixa, cada iteração faz um ajuste muito pequeno nos pesos (passo muito pequeno).

- Pode precisar de muitas iterações para convergir para o ponto de mínimo desejado na superfície de busca.

Usando taxa de aprendizado muito alta, cada iteração faz um ajuste muito grande nos pesos (passo muito grande).

- Pode causar oscilações em torno de um ponto de mínimo.

Medidas de Erros

Para ambos os tipos de problema, será usado o erro SSE (*sum squared error* - soma dos erros quadráticos).

Ex.:

	Saídas da rede			Saídas desejadas		
Padrão	1	...	N	1	...	N
Nodo 1	0.98	...	0.12	1.00	...	0.00
Nodo 2	0.02	...	0.96	0.00	...	1.00

Soma dos erros quadráticos (SSE):

$$SSE = (0.98 - 1.00)^2 + \ldots + (0.12 - 0.00)^2 + (0.02 - 0.00)^2 + \ldots + (0.96 - 1.00)^2.$$

Medidas de Erros

Para problemas de classificação, também será calculado o erro de classificação (neste projeto, só para o conjunto de teste).

Regra de classificação winner-takes-all:

O neurônio de saída que apresentar o maior valor de saída determina a classe do padrão.

Ex.:

	Saídas da rede			Saídas desejadas		
Padrão	1	...	N	1	...	N
Nodo 1	0.98	...	0.12	1.00	...	0.00
Nodo 2	0.02	...	0.96	0.00	...	1.00
Classe	1	...	2	1	...	2

$$\text{Erro Classif.} = 100 \times \frac{\text{Quant. de padrões classificados erradamente}}{\text{Quant. total de padrões}}$$

Backpropagation

Será usado o algoritmo *Backpropagation* padrão

É um algoritmo de gradiente descendente, ou seja, utiliza informações de derivada.

Por isso, as funções de ativação devem ser contínuas e diferenciáveis (é o caso da sigmóide logística).

Objetivo:

Fazer “ajuste de pesos”, ou seja, escolher os pesos que geram as saídas mais corretas possíveis (menor erro) de forma iterativa.

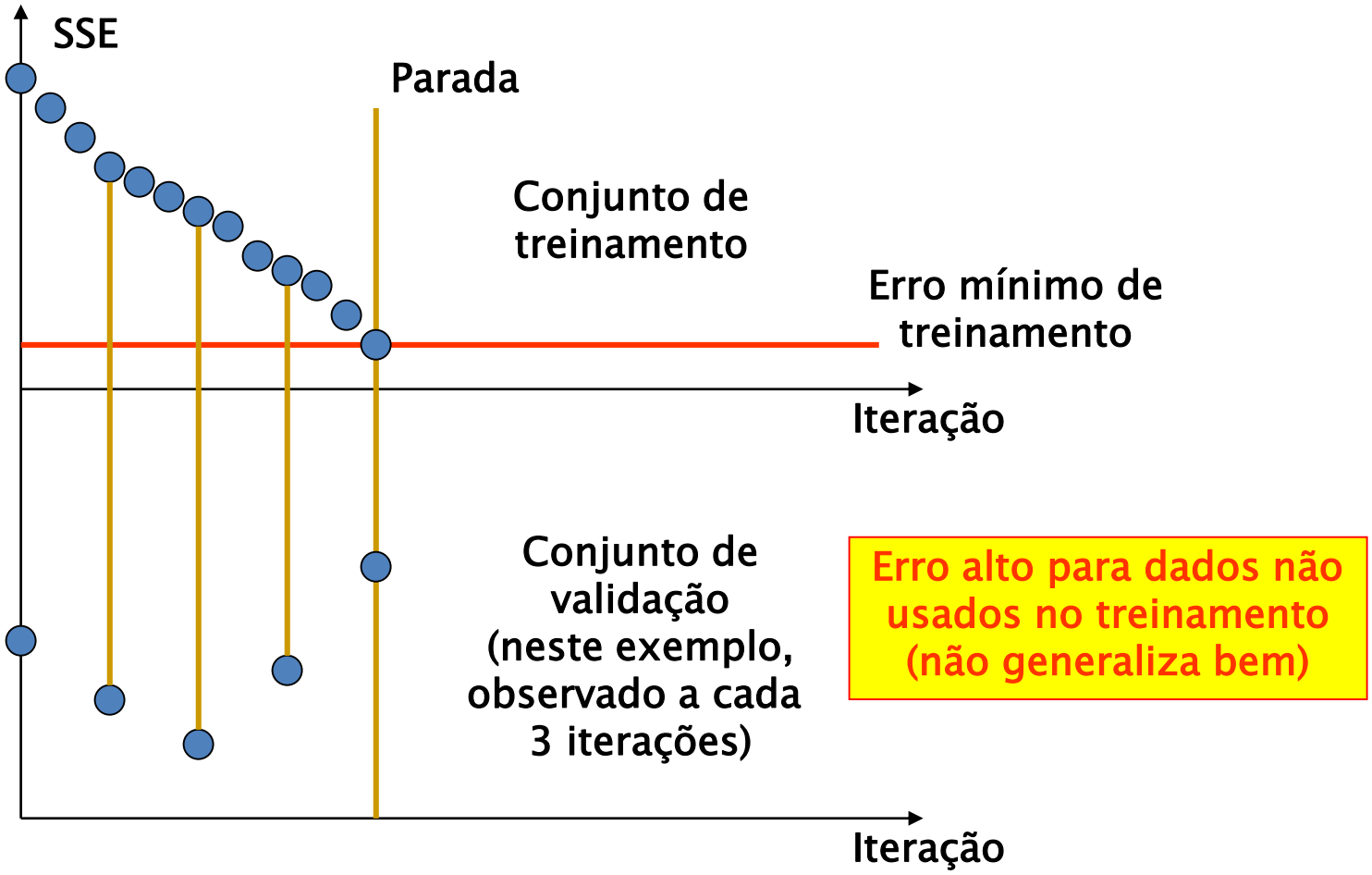
Idéia geral:

A cada iteração, obter um erro cada vez menor para os dados de treinamento.

Cuidado:

Não permitir que a rede aprenda detalhes demais do conjunto de treinamento (overfitting).

Gráfico



Parada por Erro Mínimo de Validação

É recomendável que o treinamento seja interrompido quando o erro no conjunto de validação atingir um mínimo.

- A partir deste ponto, supõe-se que a rede só aprenderia detalhes irrelevantes do conjunto de treinamento.
- O erro para dados de treinamento seria cada vez menor, mas o erro para dados novos (validação) seria cada vez mais alto.

Neste projeto, será usado o seguinte critério de parada:

- Interromper o treinamento quando o erro de validação subir por 5 iterações consecutivas.
- É o critério implementado no Matlab (parâmetro "max_fail = 5").

O que vocês vão fazer?

- Vão escolher:
  - 3 quantidades de neurônios escondidos,
  - 3 taxas de aprendizado.
- Temos um total de 9 configurações a serem testadas.
- Para cada configuração, será realizado um treinamento.
- A melhor configuração a ser escolhida é a de menor erro de teste.

Config.	SSE de Treinamento	SSE de Teste
1	2.13	3.45
2	1.44	**0.71**
...	...	...
9	4.43	5.18

O que vocês vão fazer?

- Para a melhor configuração escolhida, devem ser feitos 10 treinamentos com diferentes inicializações de pesos.
- O objetivo é verificar como a melhor rede se comporta quando variamos os pesos iniciais.

Config	SSE de Treinamento	SSE de validação	SSE de Teste	Erro de Classificação
1	2.4	2.13	3.45	0.2
2	5.9	1.44	**0.71**	0.4
...	...	...	...	1.6
10	2.64	4.43	5.18	2.0

Para começar...

```
bool_in=0
real_in=8
bool_out=2
real_out=0
training_examples=384
validation_examples=192
test_examples=192
0.352941 0.83 0.606557 0 0 0.396423 0.0964987 0.75 0 1
0.352941 0.77 0.606557 0.32 0.228132 0.436662 0.324936 0.3 0 1
0.117647 0.46 0.622951 0.2 0 0.360656 0.691716 0.116667 0 1
0.117647 0.555 0.491803 0 0 0.390462 0.113151 0.0333333 0 1
0.470588 0.325 0.590164 0.23 0 0.4769 0.222886 0.35 0 1
0.0588235 0.53 0.57377 0.28 0.159574 0.509687 0.0273271 0.0166667 0 1
0 0.49 0.672131 0.15 0.0992908 0.375559 0.0943638 0.0166667 0 1
0.235294 0.475 0.57377 0.32 0 0.47839 0.22801 0.05 0 1
0.411765 0.405 0.639344 0.4 0.0567376 0.695976 0.0781383 0.35 0 1
0.176471 0.565 0.360656 0.13 0 0.33383 0.0264731 0.0166667 0 1
0.117647 0.56 0.540984 0.22 0 0.372578 0.0977797 0.05 0 1
0.352941 0.615 0.590164 0.45 0.271868 0.500745 0.279675 0.216667 0 1
0 0.9 0.639344 0.63 0.0165485 0.885246 1 0.0666667 1 0
0.176471 0.965 0.57377 0.31 0 0.520119 0.0695986 0.0666667 1 0
0.176471 0.865 0.672131 0.48 0.549645 0.57228 0.879163 0.0666667 1 0
0.0588235 0.485 0.557377 0.21 0 0.405365 0.434244 0.0166667 0 1
0.411765 0.75 0.639344 0.29 0.148936 0.52459 0.262169 0.55 1 0
0.764706 0.725 0.672131 0.19 0.130024 0.330849 0.0713066 0.6 0 1
0.411765 0.68 0.606557 0.26 0.159574 0.387481 0.242955 0.5 0 1
0.176471 0.65 0.639344 0.23 0.0933806 0.423249 0.104611 0.216667 1 0
0.235294 0.425 0.47541 0.22 0.0579196 0.414307 0.0973527 0.116667 0 1
```

Atenção, POR FAVOR!!

```
bool_in=0
real_in=8
bool_out=2
real_out=0
training_examples=384
validation_examples=192
test_examples=192
```

Conjunto de Dados

384 linhas

Treinamento.txt

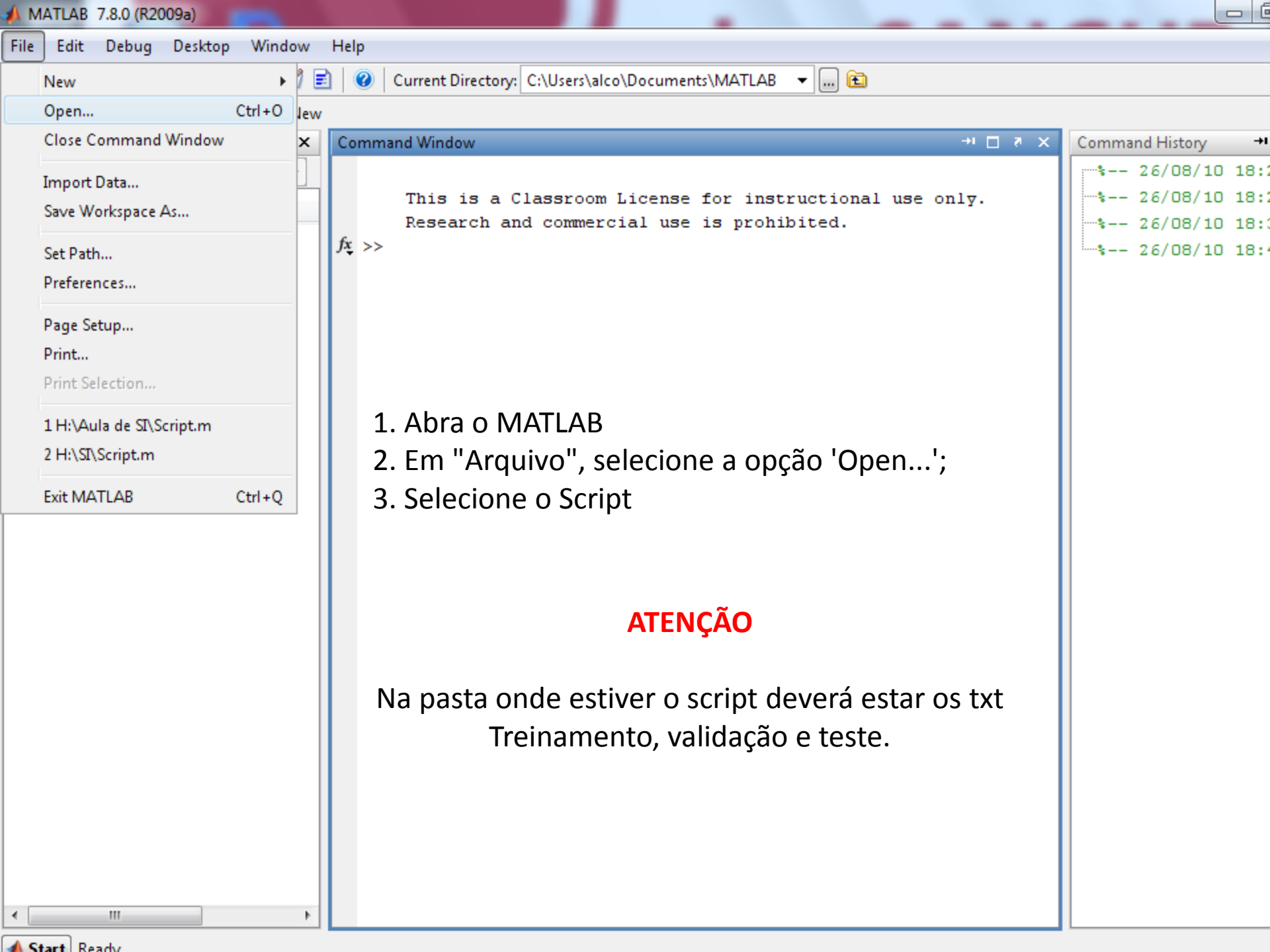
192 linhas

Validacao.txt

192 linhas

Teste.txt

```
bool_in=0
real_in=8
bool_out=2
real_out=0
training_examples=384
validation_examples=192
test_examples=192
0.352941 0.83 0.606557 0 0 0.396423 0.0964987 0.75 0 1
0.352941 0.77 0.606557 0.32 0.228132 0.436662 0.324936 0.3 0 1
0.117647 0.46 0.622951 0.2 0 0.360656 0.691716 0.116667 0 1
0.117647 0.555 0.491803 0 0 0.390462 0.113151 0.0333333 0 1
0.470588 0.325 0.590164 0.23 0 0.4769 0.222886 0.35 0 1
0.0588235 0.53 0.57377 0.28 0.159574 0.509687 0.0273271 0.0166667 0 1
0 0.49 0.672131 0.15 0.0992908 0.375559 0.0943638 0.0166667 0 1
0.235294 0.475 0.57377 0.32 0 0.47839 0.22801 0.05 0 1
0.411765 0.405 0.639344 0.4 0.0567376 0.695976 0.0781383 0.35 0 1
0.176471 0.565 0.360656 0.13 0 0.33383 0.0264731 0.0166667 0 1
0.117647 0.56 0.540984 0.22 0 0.372578 0.0977797 0.05 0 1
0.352941 0.615 0.590164 0.45 0.271868 0.500745 0.279675 0.216667 0 1
0 0.9 0.639344 0.63 0.0165485 0.885246 1 0.0666667 1 0
0.176471 0.965 0.57377 0.31 0 0.520119 0.0695986 0.0666667 1 0
0.176471 0.865 0.672131 0.48 0.549645 0.57228 0.879163 0.0666667 1 0
0.0588235 0.485 0.557377 0.21 0 0.405365 0.434244 0.0166667 0 1
0.411765 0.75 0.639344 0.29 0.148936 0.52459 0.262169 0.55 1 0
0.764706 0.725 0.672131 0.19 0.130024 0.330849 0.0713066 0.6 0 1
0.411765 0.68 0.606557 0.26 0.159574 0.387481 0.242955 0.5 0 1
0.176471 0.65 0.639344 0.23 0.0933806 0.423249 0.104611 0.216667 1 0
0.235294 0.425 0.47541 0.22 0.0579196 0.414307 0.0973527 0.116667 0 1
```


MATLAB 7.8.0 (R2009a)
File Edit Debug Desktop Window Help
New
Open... Ctrl+O
Close Command Window
Import Data...
Save Workspace As...
Set Path...
Preferences...
Page Setup...
Print...
Print Selection...
1 H:\Aula de SI\Script.m
2 H:\SI\Script.m
Exit MATLAB Ctrl+Q
Current Directory: C:\Users\alco\Documents\MATLAB
Command Window
This is a Classroom License for instructional use only.
Research and commercial use is prohibited.
>>
Command History
1. Abra o MATLAB
2. Em "Arquivo", selecione a opção 'Open...';
3. Selecione o Script
ATENÇÃO
Na pasta onde estiver o script deverá estar os txt
Treinamento, validação e teste.
Start
Ready

Editor - H:\Aula de SI\Script.m

File Edit Text Go Cell Tools Debug Desktop Window Help

Stack: Base

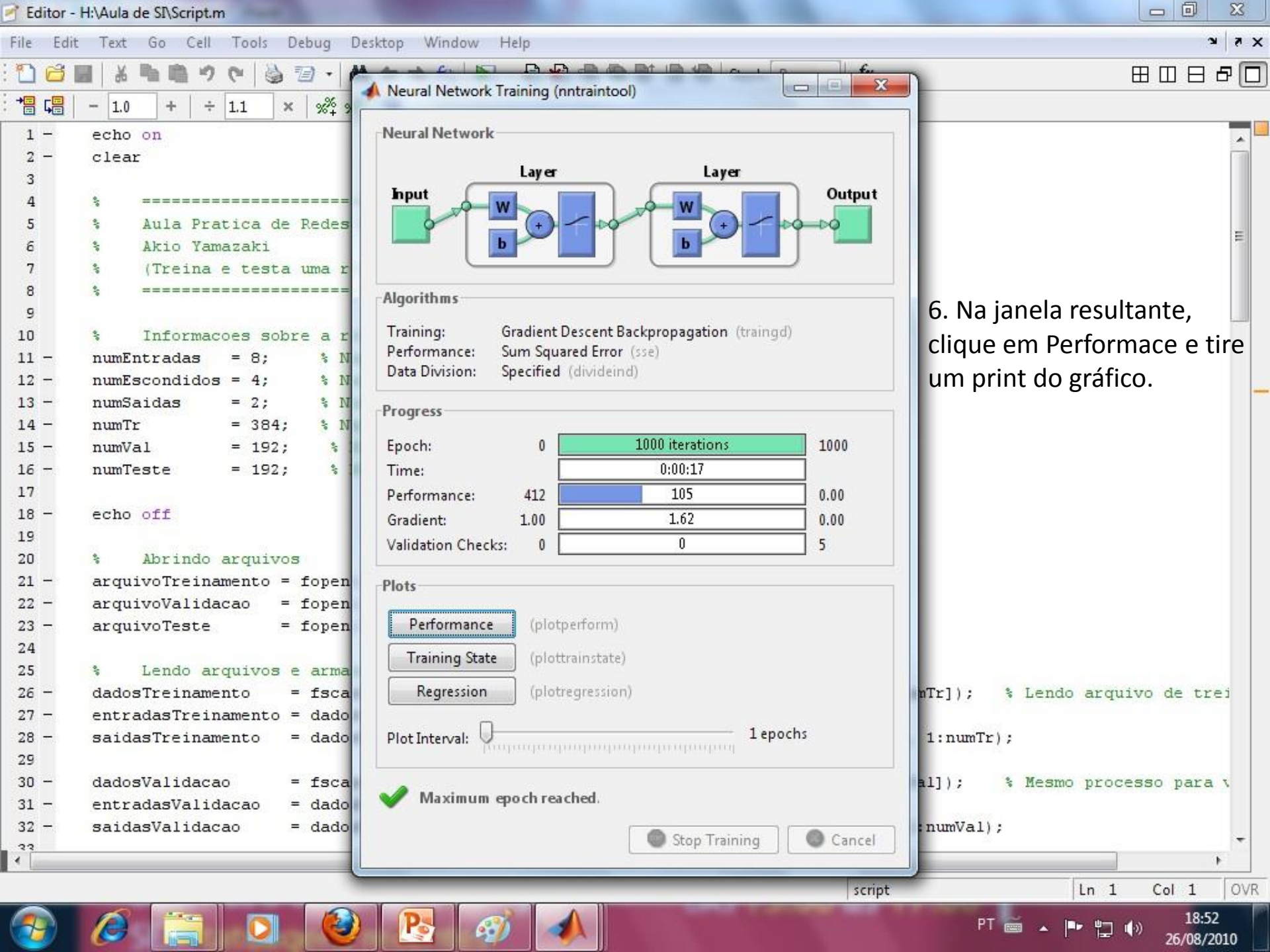
```
1  -    echo on
2  -    clear
3
4        %    ================================================================
5        %    Aula Pratica de Redes Neurais
6        %    Akio Yamazaki
7        %    (Treina e testa uma rede MLP)
8        %    ================================================================
9
10       %    Informacoes sobre a rede e os dados
11 -     numEntradas   = 8;      % Numero de nodos de entrada
12 -     numEscondidos = 4;      % Numero de nodos escondidos
13 -     numSaidas     = 2;      % Numero de nodos de saida
14 -     numTr         = 384;    % Numero de padroes de treinamento
15 -     numVal        = 192;     % Numero de padroes de validacao
16 -     numTeste      = 192;     % Numero de padroes de teste
17
18 -     echo off
19
20       %    Abrindo arquivos
21 -     arquivoTreinamento = fopen('treinamento.txt','rt');
22 -     arquivoValidacao   = fopen('validacao.txt','rt');
23 -     arquivoTeste       = fopen('teste.txt','rt');
24
25       %    Lendo arquivos e armazenando dados em matrizes
26 -     dadosTreinamento     = fscanf(arquivoTreinamento,'%f',[(numEntradas + numSaidas), numTr]);   % Lendo arquivo de trei
27 -     entradasTreinamento = dadosTreinamento(1:numEntradas, 1:numTr);
28 -     saidasTreinamento   = dadosTreinamento((numEntradas + 1):(numEntradas + numSaidas), 1:numTr);
29
30 -     dadosValidacao       = fscanf(arquivoValidacao,'%f',[(numEntradas + numSaidas), numVal]);    % Mesmo processo para v
31 -     entradasValidacao    = dadosValidacao(1:numEntradas, 1:numVal);
32 -     saidasValidacao      = dadosValidacao((numEntradas + 1):(numEntradas + numSaidas), 1:numVal);
```

4. Modifique o Script para que ele se adeque ao seu caso.

5. Rode o script

script Ln 10 Col 9 OVR

PT 18:49 26/08/2010



6. Na janela resultante, clique em Performace e tire um print do gráfico.

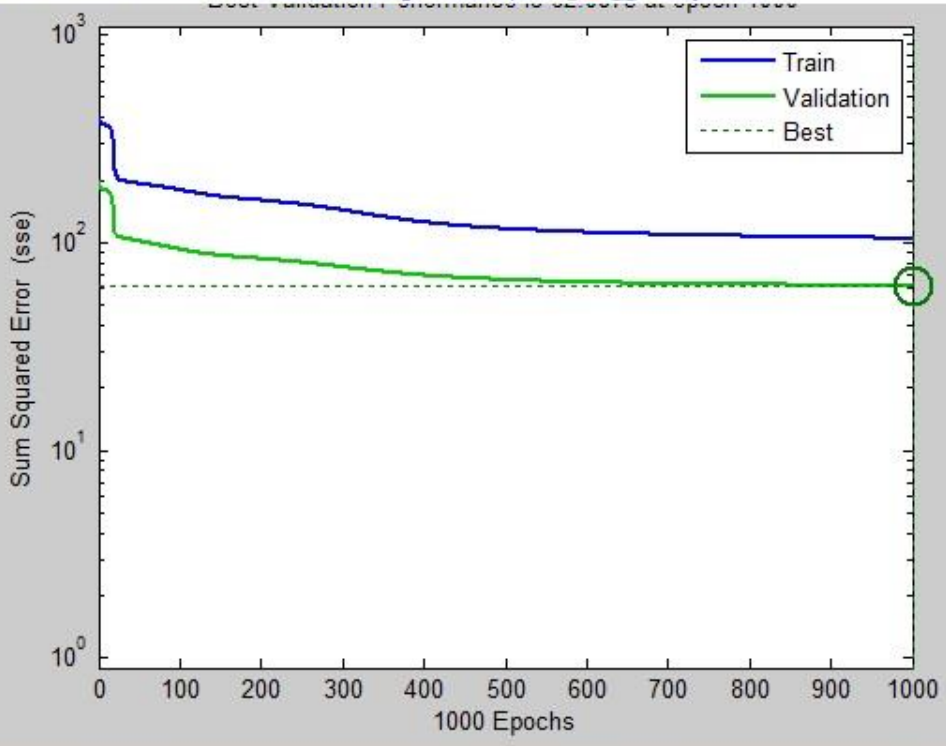
Editor - H:\Aula de SI\Script.m

File Edit Text Go Cell Tools Debug Desktop Window Help

```
1  - echo on
2  - clear
3
4  - %     ==========
5  - %     Aula Pratic
6  - %     Akio Yamaza
7  - %     (Treina e t
8  - %     ==========
9
10 - %     Informacoes
11 - numEntradas    =
12 - numEscondidos  =
13 - numSaidas      =
14 - numTr          =
15 - numVal         =
16 - numTeste       =
17
18 - echo off
19
20 - %     Abrindo arq
21 - arquivoTreinamen
22 - arquivoValidacao
23 - arquivoTeste
24
25 - %     Lendo arquivos e arma
26 - dadosTreinamento     = fsca
27 - entradasTreinamento = dado
28 - saidasTreinamento    = dado
29
30 - dadosValidacao       = fsca
31 - entradasValidacao    = dado
32 - saidasValidacao      = dado
```

Performance (plotperform)

Note new toolbar buttons: data brushing & linked plots Play video



Regression (plotregression)

Plot Interval: 1 epochs

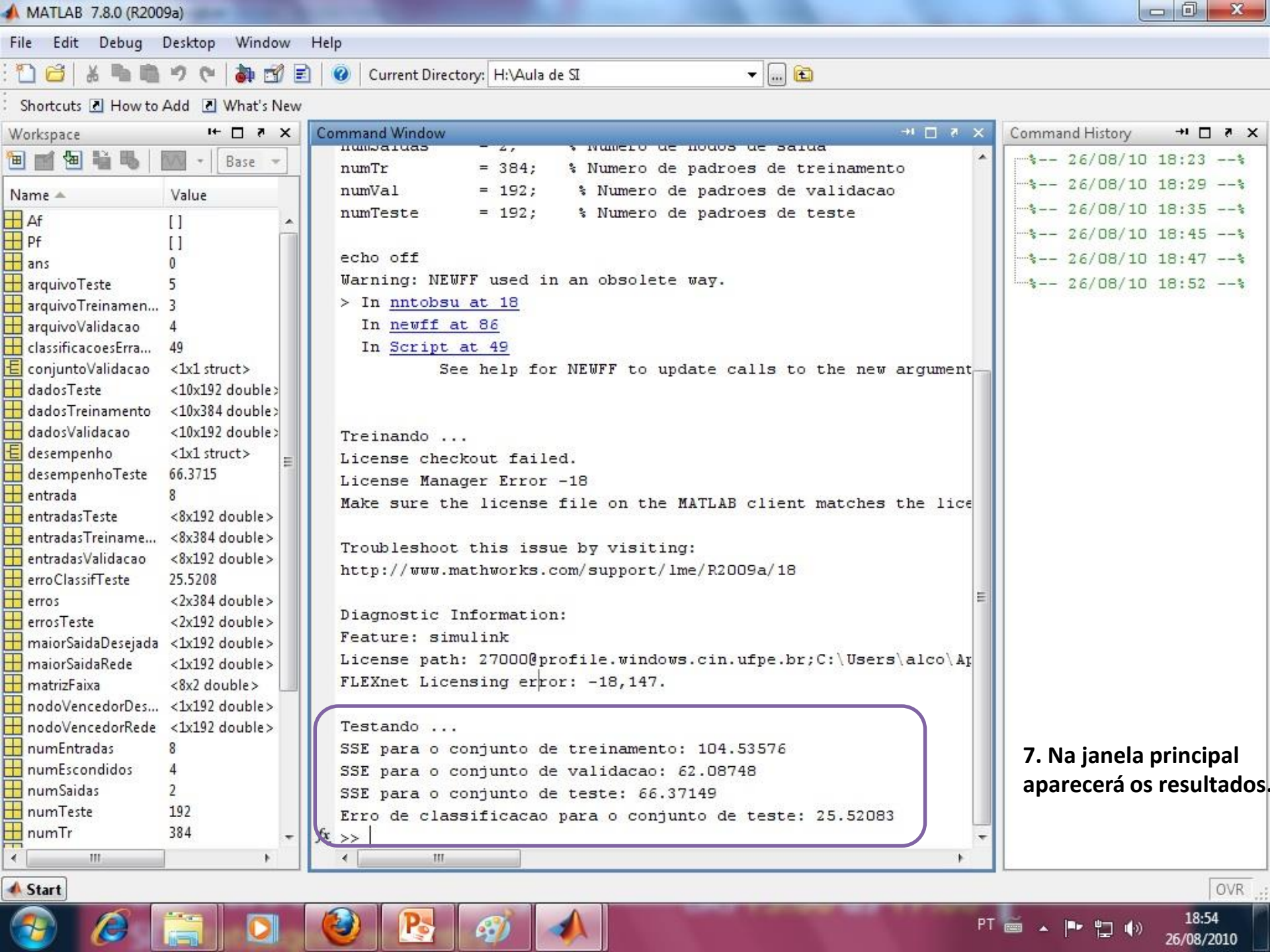
Opening Performance Plot

Stop Training Cancel

```
26 - dadosTreinamento     = fsca                    nTr]);    % Lendo arquivo de trei
27 - entradasTreinamento = dado
28 - saidasTreinamento    = dado                     1:numTr);
29
30 - dadosValidacao       = fsca                    al]);     % Mesmo processo para v
31 - entradasValidacao    = dado
32 - saidasValidacao      = dado                    :numVal);
```

script Ln 1 Col 1 OVR

PT 18:53 26/08/2010



7. Na janela principal aparecerá os resultados.

Dúvidas?

- Monitores:
  - Arley Ristar – arrr2@cin.ufpe.br
  - Thiago Miotto – tma@cin.ufpe.br