

Tópicos

Avançados II

Aula 11

FINALMENTE CRIEI MEU PRÓPRIO **MODELO NEURAL**

AUGUSTO UCHÔA

Petran- Programa de Pós-graduação em
Engenharia de Transportes



TRILHA DE HOJE:



Experimentar o potencial do Matlab para criação de modelos neurais com o uso de codificação

Usar IA generativa para auxiliar na criação de códigos

Criar um modelo neural dedicado à previsão ou classificação, usando seus dados, para fins de publicação de artigo científico

O fenômeno é da área de infraestrutura de transportes, linha de caracterização de materiais, projeto de pesquisa do DNIT.

Dispomos de um conjunto de dados referente às características geotécnicas de amostras de solos, coletadas no estado do Ceará, tanto de subleitos quanto de jazidas.

Essas características referem-se aos resultados de ensaios de laboratório tais como: granulometria, umidade ótima, densidade máx, Expansão, coordenadas utm da amostra e CBR.

Pretende-se estimar o valor do CBR (*California Bearing Ratio*) em função de sua localização e dos ensaios geotécnicos básicos.

O dataset está organizado em um arquivo excel (*.xlsx), na Planilha 1.

Nesta planilha os dados para modelagem estão organizados no range de E2:Q3930. Sendo que são 13 colunas por 3929 linhas. As variáveis dependentes(entradas) estão nas colunas de E até P e a variável dependente (saída) na coluna Q

ENTENDENDO O PROBLEMA

CONSIDERAÇÕES:

1. Como as redes neurais de apenas 1 camada intermediária não foram capazes de melhorar o resultado para um RMS maior que 80%, optou-se por tentar redes com duas camadas;
2. Contudo, o APP nftool não permite o uso de mais de uma camada, neste caso, existem duas opções:
 - a) exportar o código do modelo gerado pelo app, para a janela de comando e alterá-lo com a ajuda de IA generativa ou,
 - b) Criar um modelo do zero, gerando o código dele com IA generativa.

CARREGANDO E PRÉ- PROCESSANDO

```
% Carregar o arquivo excel
dataset = xlsread('solos_dnit.xlsx', 1, 'E2:Q3930');

% Criar uma ordem randômica das 3930 linhas
randomOrder = randperm(size(dataset, 1));
dataset = dataset(randomOrder, :);

% Normalizar todos os valores usando a função mini-max entre 0 e 1
colsToNormalize = 1:size(dataset, 2);
colsToNormalize(colsToNormalize == size(dataset, 2)) = [];
colsToNormalize(colsToNormalize == 17) = [];
dataset(:, colsToNormalize) = (dataset(:, colsToNormalize) - min(dataset(:, colsToNormalize))) ./ (max(dataset(:, colsToNormalize)) - min(dataset(:, colsToNormalize)));

% Separar em 2 conjuntos de dados mutuamente exclusivos
numRows = size(dataset, 1);
numTrain = round(0.7 * numRows); % 70% dos dados para treinamento

% Conjunto de treinamento
treinamento = dataset(1:numTrain, :);

% Conjunto de teste
teste = dataset(numTrain+1:end, :);

% Dividir cada conjunto em dois subconjuntos
entradastreino = treinamento(:, 1:end-1); % Colunas E até P
saidastreino = treinamento(:, end); % Coluna Q

entradasteste = teste(:, 1:end-1); % Colunas E até P
saidateste = teste(:, end); % Coluna Q
```



OPÇÃO a

1. O carregamento do arquivo e pré-processamento é o mesmo da aula de anterior;
2. Trata-se de um problema de regressão
3. Usaremos o modelo de uma camada gerado no nftool para a janela de comando do Matlab e o alteraremos visando criar um modelo com 2 camadas intermediárias

```
% saidatreinamento' target data:  
x = entradastreinamento';  
t = saidatreinamento';
```

```
% Choose a Training Function  
% For a list of all training functions type: help nntrain  
% 'trainlm' is usually fastest.  
% 'trainbr' takes longer but may be better for challenging problems.  
% 'trainscg' uses less memory. Suitable in low memory situations.  
trainFcn = 'trainlm'; % Levenberg-Marquardt backpropagation.
```

```
% Create a Fitting Network with 2 hidden layers, each with 13 neurons  
hiddenLayerSize = [13, 13];  
net = fitnet(hiddenLayerSize,trainFcn);
```

```
% Choose Input and Output Pre/Post-Processing Functions  
% For a list of all processing functions type: help nnprocess  
net.input.processFcns = {'removeconstantrows','mapminmax'};  
net.output.processFcns = {'removeconstantrows','mapminmax'};
```

```
% Setup Division of Data for Training, Validation, Testing  
% For a list of all data division functions type: help nndivision  
net.divideFcn = 'dividerand'; % Divide data randomly  
net.divideMode = 'sample'; % Divide up every sample  
net.divideParam.trainRatio = 70/100;  
net.divideParam.valRatio = 15/100;  
net.divideParam.testRatio = 15/100;
```

```
% Choose a Performance Function  
% For a list of all performance functions type: help nnperformance  
net.performFcn = 'mse'; % Mean Squared Error
```

No Bing CHAT

“Atue como um especialista em Bing CHAT e Matlab e altere o seguinte código, para que seja possível treinar a rede com 2 camadas ocultas, de 13 neurônios em cada camada.”

Claro, você pode modificar a linha que cria a rede neural para ter duas camadas ocultas, cada uma com 13 neurônios. Aqui está como você pode fazer isso:

```
% Crie uma rede de ajuste com 2 camadas ocultas, cada  
uma com 13 neurônios  
hiddenLayerSize = [13, 13];  
net = fitnet(hiddenLayerSize,trainFcn);
```

MATLAB R2023a - academic use

HOME PLOTS APPS EDITOR PUBLISH VIEW

Search Documentation Carlos

FILE NAVIGATE CODE ANALYZE SECTION RUN

Run Step Stop

C:\Users\ACER\Desktop\dados dnit

Current Folder

Editor - C:\Users\ACER\Desktop\dados dnit\terceiro.m

```

10 x = entradatreinamento';
11 t = saidatreinamento';
12
13 % Choose a Training Function
14 % For a list of all training functions type: help nntrain
15 % 'trainlm' is usually fastest.
16 % 'trainbr' takes longer but may be better for challenging problems.
17 % 'trainscg' uses less memory. Suitable in low memory situations.
18 trainFcn = 'trainlm'; % Levenberg-Marquardt backpropagation
19
20 % Create a Fitting Network with 2 hidden layers, each with 13 neurons
21 hiddenLayerSize = [13, 13];
22 net = fitnet(hiddenLayerSize,trainFcn);
23
24 % Choose Input and Output Pre/Post-Processing Functions
25 % For a list of all processing functions type: help nnprocess
26 net.input.processFns = {'removeconstantrows','mapminmax'};
27 net.output.processFns = {'removeconstantrows','mapminmax'};
28
29 % Setup Division of Data for Training, Validation, Testing
30 % For a list of all data division functions type: help nnpartition

```

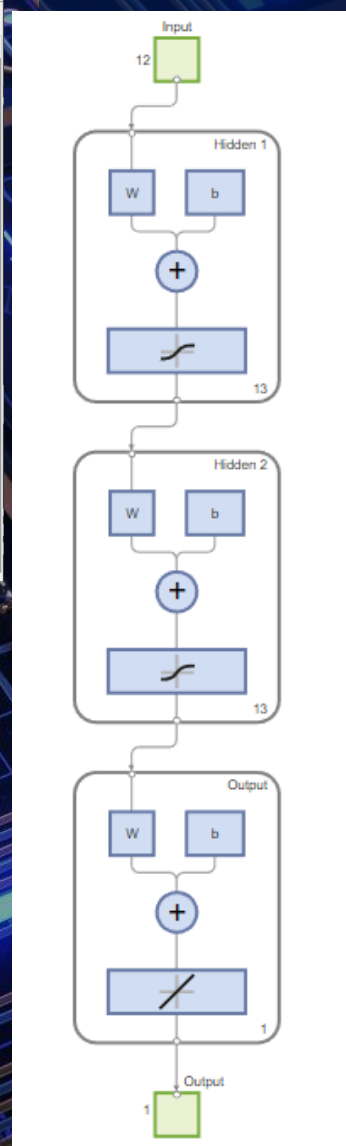
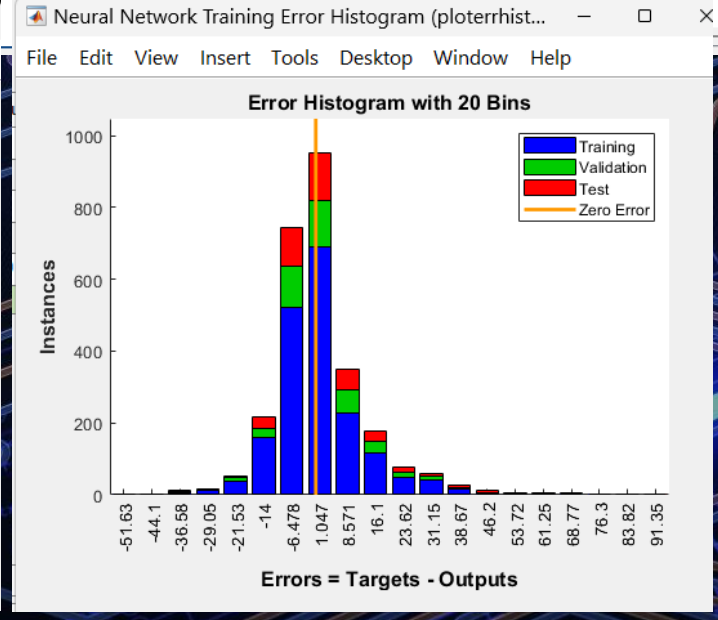
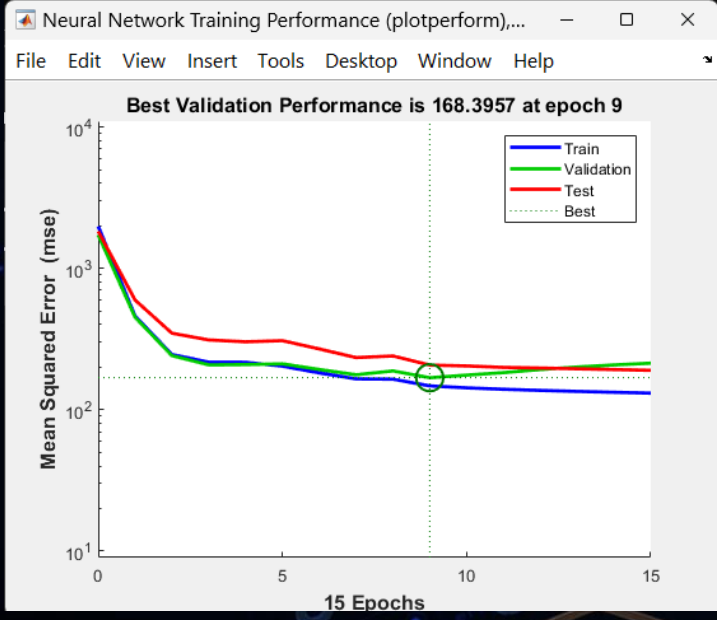
Workspace

Name	Value
colsToNormalize	1x12 double
dataset	3929x13 double
e	1x2750 double
entradasteste	1179x12 double
entradatreinamento	2750x12 double
hiddenLayerSize	[13,13]
net	1x1 network
numRows	3929
numTrain	2750
performance	159.5634
randomOrder	1x3929 double
saidateste	1179x1 double
saidatreinamento	2750x1 double
t	1x2750 double
teste	1179x13 double
testPerformance	206.9540
testTargets	1x2750 double
tr	1x1 struct

Command Window

```
testPerformance =
    206.9540
```

NO MATLAB



Neural Network Training (03-Nov-2023) ...

Network Diagram

Training Results

Training finished: Met validation criterion ✓

Training Progress

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	15	1000
Elapsed Time	-	00:00:04	-
Performance	1.99e+03	131	0
Gradient	1.39e+04	106	1e-07
Mu	0.001	1	1e+10
Validation Checks	0	6	6

Training Algorithms

Data Division: Random dividerand
 Training: Levenberg-Marquardt trainlm
 Performance: Mean Squared Error mse
 Calculations: MEX

Training Plots

Performance Training State
 Error Histogram Regression
 Fit

VERSÕES DO MATLAB

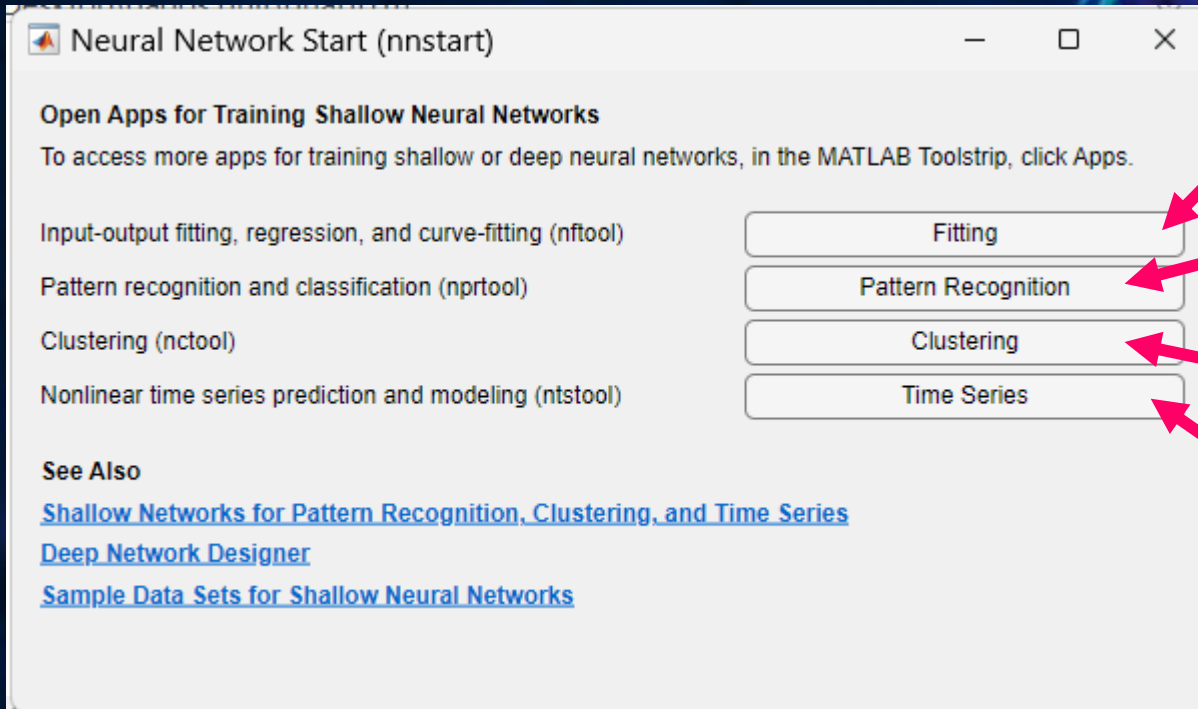


O **nntool** (até 2010) é uma interface gráfica de usuário (GUI) mais antiga para redes neurais. Ele permite que você crie, treine e teste redes neurais em um ambiente visual.

O **nnstart** (2010-2023) é uma ferramenta mais recente que também fornece uma GUI para trabalhar com redes neurais. Ele inclui várias outras GUIs, como o 'neural network fitting app', que são mais recentes e, portanto, recomendadas

O **nntool** e o **nnstart** são duas ferramentas diferentes disponíveis em diferentes versões do MATLAB para trabalhar com redes neurais. Ambos podem ser usados para trabalhar com redes neurais, mas o nnstart é mais recente e pode oferecer mais recursos.

NO MATLAB 2023



- Para Regressão/previsão/estimação (MLP)
- Para Classificação/reconhecimento de padrões (MLP)
- Para Clusterização/agrupamentos (Mapas de Kohonen)
- Para estimação baseada em séries temporais (redes recorrentes)

Já foi desenvolvido sob a transição de redes neurais clássicas para profundas e já exibem algumas características de ambas. Podem ser usados de forma análoga ao como o nprtool e o nftool.

OPÇÃO b

1. O carregamento do arquivo e pré-processamento é o mesmo da aula de anterior;
2. Trata-se de um problema de regressão
3. Usaremos uma IA generativa e construiremos o modelo do zero, de forma a atender a todas as nossas necessidades, por exemplo, usar 2 camadas intermediárias, por ex, realizar a validação cruzada k-fold, por exemplo alterar a função de ativação das camadas intermediárias e tudo mais que desejar

Este código assume que você tem o arquivo 'dados_dnit.xlsx' no mesmo diretório do seu script MATLAB. Ele carrega os dados do arquivo Excel, randomiza as linhas da matriz de dados, normaliza as 12 primeiras colunas para o intervalo [-1, 1] usando a função min-max, e então divide os dados em dois conjuntos: 'treinamento' contendo 75% dos dados e 'teste' contendo os 25% restantes. Por favor, ajuste o código conforme necessário para o seu caso de uso específico.

```
% Randomizar as linhas da matriz
rng('default'); % Para reprodutibilidade
dados = dados(randperm(size(dados, 1)), :);

% Normalizar os dados entre -1 e 1 usando a função min-max
min_val = min(dados(:, 1:12), [], 1);
max_val = max(dados(:, 1:12), [], 1);
dados(:, 1:12) = -1 + 2 * (dados(:, 1:12) - min_val) ./ (max_val - min_val);

% Particionar a matriz em conjuntos de treinamento e teste
num_rows = size(dados, 1);
cutoff = round(0.75 * num_rows); % Índice de corte para 75% dos dados
treinamento = dados(1:cutoff, :);
teste = dados(cutoff+1:end, :);
```



CRIE UM CÓDIGO PARA MATLAB QUE USE OS FOLDS GERADOS NO CÓDIGO ANTERIOR PARA TREINAR, VALIDAR E TESTAR UMA REDE NEURAL, COM OS SEGUINTE HIPERPARÂMETROS: ALGORITMO BACKPROPAGATION LM, FUNÇÃO DE ATIVAÇÃO TANSIG NOS NEURÔNIOS DAS CAMADAS INTERMEDIÁRIAS E LINEAR NAS CAMADAS DE ENTRADA E SAÍDA. USE COMO CRITÉRIO DE PARADA O QUE FOR MAIS ADEQUADO. PLOTE O GRÁFICO DE PERFORMANCE, OS RESULTADOS FINAIS DA REDE, A MATRIZ DE CONFUSÃO O A CURVA ROC. A REDE DEVE POSSUIR 12 NEURÔNIOS NA CAMADA DE ENTRADA, 2 CAMADAS INTERMEDIÁRIAS COM 13 NEURÔNIOS EM CADA UMA DELAS E 1 NEURÔNIO NA CAMADA DE SAÍDA



PROMPT

"Considerando que possui um conjunto de dados armazenados em formato tabular, em uma planilha do excel, que devem ser usado para criação de um modelo neural. Tal arquivo se chama dados_dnit.xlsx, a planilha deste arquivo é a Planilha 1 e o range dos dados é E2:Q3930. Crie um código para o Matlab que carregue esses dados numa matriz, randomize as linhas dessa matriz, normalize entre -1 e 1 usando a função mini-max os dados contidos nas 12 primeiras colunas da matriz e depois particione essa matriz em outras duas, uma delas deve conter 75% dos dados mutuamente exclusivos e deve ser nomeada de treinamento e a outra com os 25% restante dos dados deve ser nomeada de teste."



Você deve já ter baixado e instalado o toolbox de redes neurais → neural_network_toolbox nas versões mais antigas ou o deep learning toolbox nas versões mais atuais. Baixe e instale direto da página da Matworks:

[Requisitos do produto e disponibilidade da plataforma para o Deep Learning Toolbox - MATLAB \(mathworks.com\)](#)

Para simplificar usaremos apenas a primeira linha do código², a fim de carregar no Matlab os dados do arquivo excel

```
% Carregar os dados do arquivo Excel
dados = xlsread('solos_dnit.xlsx', 1, 'E2:Q3930');
```

PRECISO QUE VOCÊ ESCREVA OUTRO CÓDIGO PARA IMPLEMENTAR A TÉCNICA DE VALIDAÇÃO CRUZADA K-FOLD NESSES DADOS, PARTICIONANDO

```
% Definir o número de folds
k = 5;

% Obter o número total de amostras
num_samples = size(dados, 1);

% Calcular o número de amostras por fold
num_samples_per_fold = floor(num_samples / k);

% Inicializar as matrizes de treinamento e teste
treinamento = cell(k, 1);
teste = cell(k, 1);

% Loop para particionar os dados em k folds
for i = 1:k
    % Índices de teste para o fold atual
    indices_teste = (1+(i-1)*num_samples_per_fold) : (i*num_samples_per_fold);

    % Índices de treinamento são todos os outros
    indices_treinamento = setdiff(1:num_samples, indices_teste);

    % Particionar os dados
    treinamento{i} = dados(indices_treinamento, :);
    teste{i} = dados(indices_teste, :);
end
```

CRIE UM CÓDIGO PARA MATLAB QUE USE OS FOLDS GERADOS NO CÓDIGO ANTERIOR PARA TREINAR, VALIDAR E TESTAR UMA REDE NEURAL, COM OS SEGUINTE HIPERPARÂMETROS: ALGORITMO BACKPROPAGATION LM, FUNÇÃO DE ATIVAÇÃO TANSIG NOS NEURÔNIOS DAS CAMADAS INTERMEDIÁRIAS E LINEAR NAS CAMADAS DE ENTRADA E SAÍDA. USE COMO CRITÉRIO DE PARADA O QUE FOR MAIS ADEQUADO. PLOTE O GRÁFICO DE PERFORMANCE, OS RESULTADOS FINAIS DA REDE, A MATRIZ DE CONFUSÃO O A CURVA ROC. A REDE DEVE POSSUIR 12 NEURÔNIOS NA CAMADA DE ENTRADA, 2 CAMADAS INTERMEDIÁRIAS COM 13 NEURÔNIOS EM CADA UMA DELAS E 1 NEURÔNIO NA CAMADA DE SAÍDA

```
% Definir a arquitetura da rede neural
hiddenLayerSize = [13 13];
net = fitnet(hiddenLayerSize);

% Definir o número de neurônios na camada de entrada
net.inputs{1}.size = 12;

% Treinar a rede neural para cada fold
for i = 1:k
    % Obter os dados de treinamento e teste para o fold atual
    treinamento_atual = treinamento{i};
    teste_atual = teste{i};

    % Separar os dados de entrada e saída
    entradas_treinamento = treinamento_atual(:, 1:end-1);
    saidas_treinamento = treinamento_atual(:, end);

    % Treinar a rede neural
    net = train(net, entradas_treinamento', saidas_treinamento');

    % Testar a rede neural
    entradas_teste = teste_atual(:, 1:end-1);
    predicoes = net(entradas_teste');

    % Calcular e exibir o erro quadrático médio
    mse = mean((predicoes - teste_atual(:, end)).^2);
    disp(['MSE para o fold ' num2str(i) ': ' num2str(mse)]);
end
```

HOME PLOTS APPS EDITOR PUBLISH VIEW

New Open Save Compare Print Go To Find Bookmark Refactor Profiler Analyze Run Section Run and Advance Run Step Stop

FILE NAVIGATE CODE ANALYZE SECTION RUN

C:\Users\ACER\Desktop\dados dnit

Current Folder

Name

- terceiro.m
- solos_dnit.xlsx
- segundo.m
- quarto.m
- net.mat

```

57 % Obter os dados de treinamento e teste para o fold atual
58 treinamento_atual = treinamento{i};
59 teste_atual = teste{i};
60
61 % Separar os dados de entrada e saída
62 entradas_treinamento = treinamento_atual(:, 1:end-1);
63 saidas_treinamento = treinamento_atual(:, end);
64
65 % Treinar a rede neural
66 net = train(net, entradas_treinamento', saidas_treinamento');
67
68 % Testar a rede neural
69 entradas_teste = teste_atual(:, 1:end-1);
70 predicoes = net(entradas_teste');
71
72 % Calcular e exibir o erro quadrático médio
73 mse = mean((predicoes - teste_atual(:, end)).^2);
74 disp(['MSE para o fold ' num2str(i) ': ' num2str(mse)]);
75 end
76

```

Workspace

Name	Value
cutoff	2947
dados	3929x13 double
entradas_teste	785x12 double
entradas_treinamento	3144x12 double
hiddenLayerSize	[13,2]
i	5
indices_teste	1x785 double
indices_treinamento	1x3144 double
k	5
max_val	1x12 double
min_val	1x12 double
mse	NaN
net	1x1 network
num_rows	3929
num_samples	3929
num_samples_per_fold	785
predicoes	1x785 double
saidas_treinamento	3144x1 double
teste	5x1 cell

Select a file to view data

Command Window

```

MSE para o fold 3: NaN
MSE para o fold 4: NaN
MSE para o fold 5: NaN
>> nnstart
fx >>

```

```
% Definir a arquitetura da rede neural
hiddenLayerSize = [13 2];
net = fitnet(hiddenLayerSize);
```

Network Diagram

Training Results

Training finished: Met validation criterion ✔

Training Progress

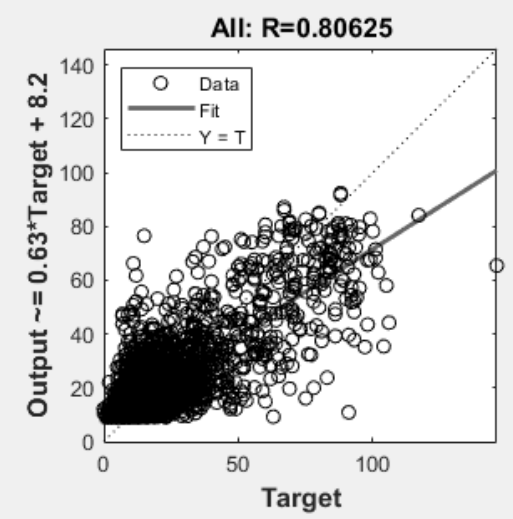
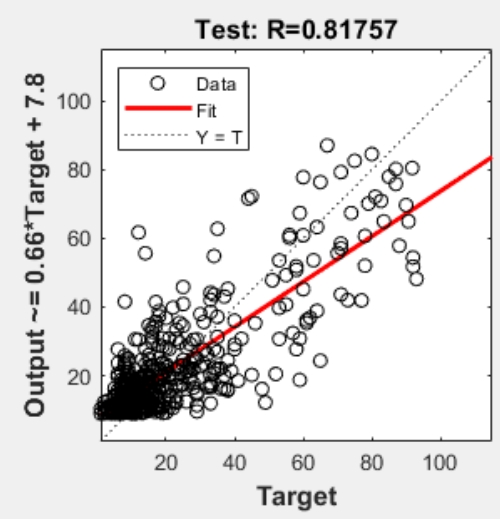
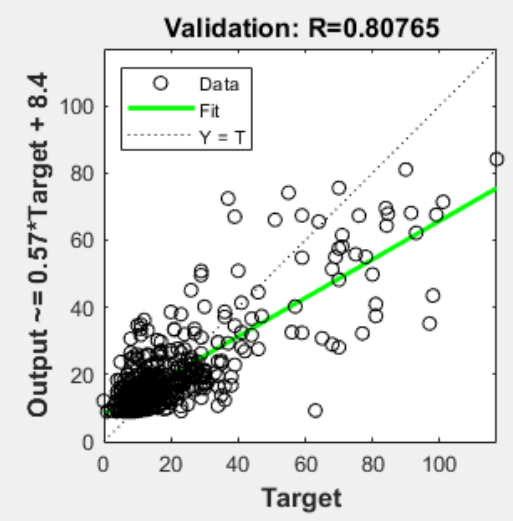
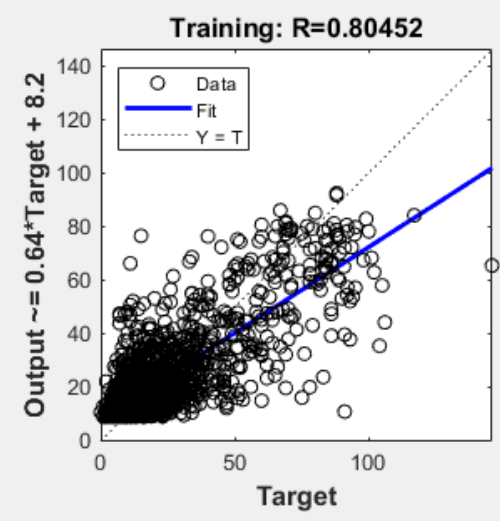
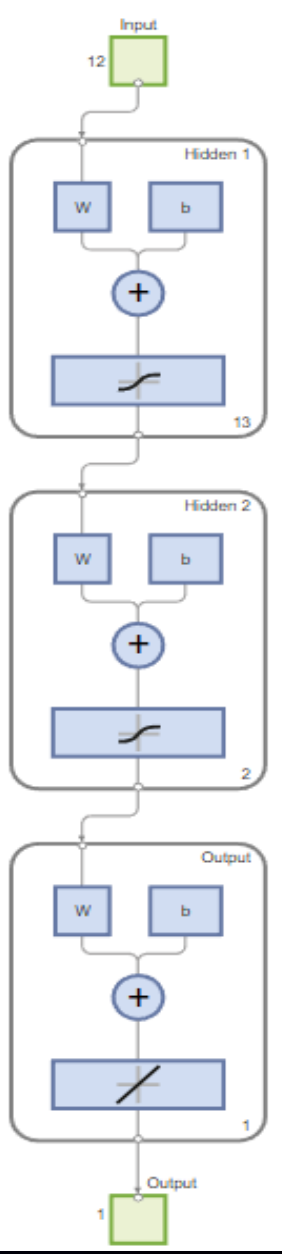
Unit	Initial Value	Stopped Value	Target Value
Epoch	0	6	1000
Elapsed Time	-	00:00:01	-
Performance	131	108	0
Gradient	353	141	1e-07
Mu	0.001	0.001	1e+10
Validation Checks	0	6	6

Training Algorithms

Data Division: Random dividerand
 Training: Levenberg-Marquardt trainlm
 Performance: Mean Squared Error mse
 Calculations: MEX

Training Plots

- Performance
- Training State
- Error Histogram
- Regression
- Fit



NÃO TÁ SATISFEITO? QUER MUDAR A ESTRUTURA DA REDE? A FUNÇÃO? O CRITÉRIO DE PARADA? O ALGORITMO? BASTA ALTERAR O CÓDIGO

ALTERE A ARQUITETURA DA REDE

REDE 12: 13:2:1 →

```
% Definir a arquitetura da rede neural  
hiddenLayerSize = [13 2];  
net = fitnet(hiddenLayerSize);
```

REDE 12: 25:2:1 →

```
% Definir a arquitetura da rede neural  
hiddenLayerSize = [25 2];  
net = fitnet(hiddenLayerSize);
```

Obs: Note que mesmo fazendo esse refinamento, os resultados não melhoram muito e o RMS chega a no máximo 80% para o conjunto de teste...isso significa que apesar de todo o esforço de modelagem, caso tenha sido levada a cabo um extensa árvore de treinamento, os resultados não melhoram...é a hora de olhar para os dados, excluir seletivamente algumas variáveis para ver se os resultados melhoram, sugiro fazer isso com apenas 1 camada intermediária...depois disso, se não melhorar, significa que essas variáveis de entrada são insuficientes para esse modelo, seria necessário encontrar outras variáveis, que melhor expliquem o fenômeno, adicionando às que melhor se saíram nesta etapa e excluindo aquelas que não parecem influenciar no desempenho do modelo...Modelar é uma arte.

Tópicos Avançados II

Augusto Uchôa

Petrans - Programa de Pós-graduação em Engenharia de Transportes

MODELAGEM DE FENÔMENOS: Precisamos falar sobre isso!

Augusto Uchôa

From Natural Computing to Deep Learning

Tópicos Avançados II

Aula 2
Augusto Uchôa

Petrans - Programa de Pós-graduação em Engenharia de Transportes

ARTIFICIAL NEURAL NETWORKS

Tópicos Avançados II

Aula 3
Augusto Uchôa

Petrans - Programa de Pós-graduação em Engenharia de Transportes

REDES MULTICAMADAS-MLP

Tópicos Avançados II

Aula 4
Augusto Uchôa

Petrans - Programa de Pós-graduação em Engenharia de Transportes

COMO UMA REDE NEURAL MLP APRENDE?

Augusto Uchôa

Tópicos Avançados II Aula 5

Petrans - Programa de Pós-graduação em Engenharia de Transportes

Tópicos Avançados II Aula 6

DATA SCIENCE & MODELAGEM EM TRANSPORTES

AUGUSTO UCHÔA

Petrans - Programa de Pós-graduação em Engenharia de Transportes

CHATGPT COMO CO-PILOTO NA MODELAGEM NEURAL

AUGUSTO UCHÔA

Tópicos Avançados II Aula 7

Petrans - Programa de Pós-graduação em Engenharia de Transportes

Tópicos Avançados II Aula 8

FRAMEWORKS & TOOLS TO MACHINE LEARNING

AUGUSTO UCHÔA

Petrans - Programa de Pós-graduação em Engenharia de Transportes

Tópicos Avançados II Aula 9

WHAT IS YOUR MOTIVATION WITH NEURAL NETWORKS?

AUGUSTO UCHÔA

Petrans - Programa de Pós-graduação em Engenharia de Transportes

Tópicos Avançados II Aula 10

NEURAL NETWORKS FOR REGRESSION PROBLEMS

AUGUSTO UCHÔA

Petrans - Programa de Pós-graduação em Engenharia de Transportes

Tópicos Avançados II Aula 11

FINALMENTE CRIEI MEU PRÓPRIO MODELO NEURAL

AUGUSTO UCHÔA

Petrans - Programa de Pós-graduação em Engenharia de Transportes

Lembrem!

Complexidade não precisa ser complicado

Existem as Redes neurais Recorrentes, Convolucionais, mapas de Kohonen, redes Pré-treinadas e outras

A chave de uma boa modelagem reside no feeling do modelador, na qualidade, escala e representatividade dos dados e na correta aplicação da técnica de modelagem

Dominar a origem, conceitos e bases das técnicas é fundamental para a correta aplicação

NN são poderosas mas não servem para tudo

Existem as técnicas de aprendizagem clássica e as de DL também

A estatística pode e deve ser usada para auxiliar nas escolhas de variáveis

Conhecer e usar as ferramentas generativas como apoio ao processo de modelagem e análise é uma necessidade



ATIVIDADE FINAL (60%)

A ideia é realizar o fechamento desse curso, com a escrita de um artigo científico, com vistas à submissão em periódico nacional.

Eleja um periódico nacional que possua o escopo adequado à sua investigação, ou algum multidisciplinar, desde que seja classificado como B2, B1, A3, A2 ou A1 no qualis da CAPES e, use a formatação indicada por ele em seu artigo.

Para fins de lançamento de notas no SIGAA e 1ª revisão do artigo, ele deve ser postado em seu portfólio no classroom

IMPRETERIVELMENTE até o dia

15/12/2023.



SOBRE O ARTIGO:

1. O artigo deve ter cunho metodológico, dedicado à criação de um modelo neural para estimação ou previsão de um determinado fenômeno (o seu, é claro).
2. Para a escrita do artigo devem ser usadas TODAS as atividades solicitadas anteriormente: (Reflexão fundamentada sobre as características do fenômeno, o estado da arte de sua modelagem, uma análise exploratória das variáveis envolvidas, o seu dataset, sobre RNA MLP). Para apresentação e análise de resultados você deve usar os resultados de um modelo baseado rede neural do tipo MLP, a ser criado por você nesta atividade.
3. Você pode criar (treinar/validar/testar) seu modelo usando um APP no Matlab ou usar a codificação em m, auxiliado por IA generativa.
4. O artigo deve estar completo, escrito em português ou inglês: Título, abstract, resumo, palavras chave.....referências.
5. Serei o 1º co-autor do seu artigo, seu orientador, se quiser participar com ideias e revisão pode ser o 2º co-autor.
6. Em nenhuma hipótese o artigo deve ser submetido antes de ser enviado a mim nesta atividade e passar por minha prévia revisão e devolução a você para correções.
7. Por favor, releia e corrija seu texto com a ajuda de IA generativa (ortografia, pontuação, acentuação, sintaxe e demais erros triviais não devem ser enviados a mim, faça o chatbot corrigir antes de me enviar).



Até mais!

uchoa@det.ufc.br

<https://www.augustouchoa.com.br/>

lattes.cnpq.br/8545122205936411

<https://orcid.org/0000-0003-1719-799X>

<https://www.youtube.com/c/AugustoUchôa13>