Tópicos

Avançados II

Aula 10

NEURAL NETWORKS FOR REGRESSION PROBLEMS

AUGUSTO UCHÔA

Petran- Programa de Pós-graduação em Engenharia de Transportes

TRILHA DE HOJE:

- Compreender a classe de problema de PREDIÇÃO/ESTIMAÇÃO/REGRESSÃO
- Conhecer e explorar oO APP nftool do Matlab
- Criar um modelo neural dedicado à previsão de CBR de solos do estado do Ceará e analisar os resultados
- Estruturar uma árvore de modelagem dedicada à definição da arquitetura e hiperparâmetros mais adequados à geração do modelo

UMA ÚLTIMA QUESTÃO PRECISA SER RESPONDIDA!

1- Meu fenômeno é complexo, multivariado e não linear?

2-Tenho um Dataset do fenômeno? Quantos padrões o compõem?

3-Fiz uma análise estatística que correlacionem as entradas às saídas?

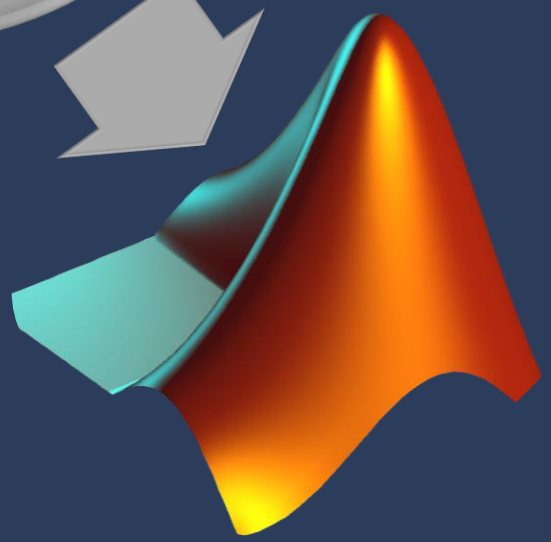
4-Fiz uma análise de colinearidade para saber se preciso usar mesmo todas as variáveis que possuo no meu Dataset?

5-Fiz um pré-processamento dos meus dados? (limpeza, organização, normalização, randomização)

6-Vou usar a técnica da validação cruzada? Se sim, quantos Folds? Já dividi meu dataset para aplicação dessa técnica?

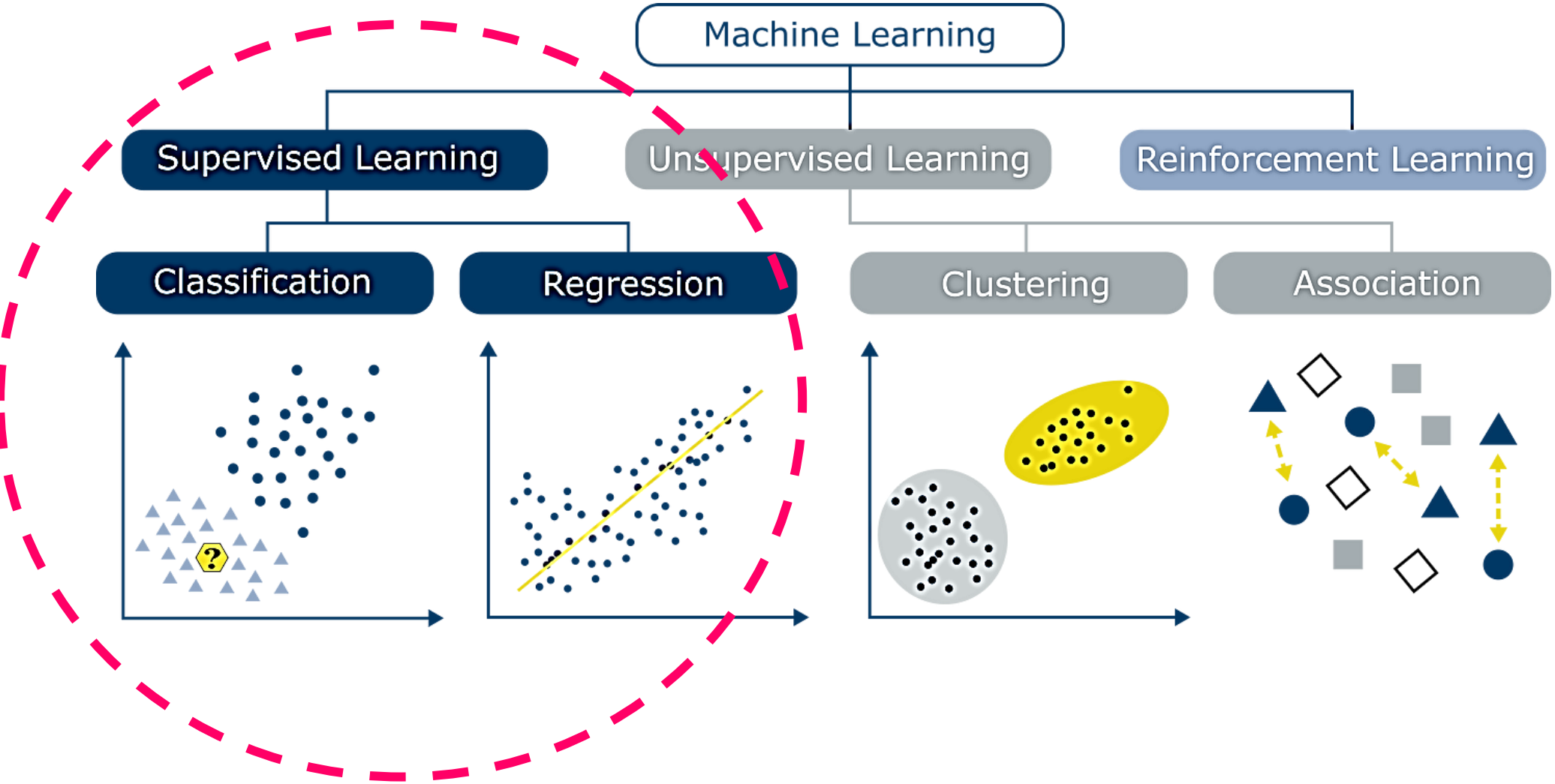
7-Vou usar alguma técnica de regularização de dados? (evitar Overfitting) se sim, qual?

8-Qual meu objetivo na modelagem?

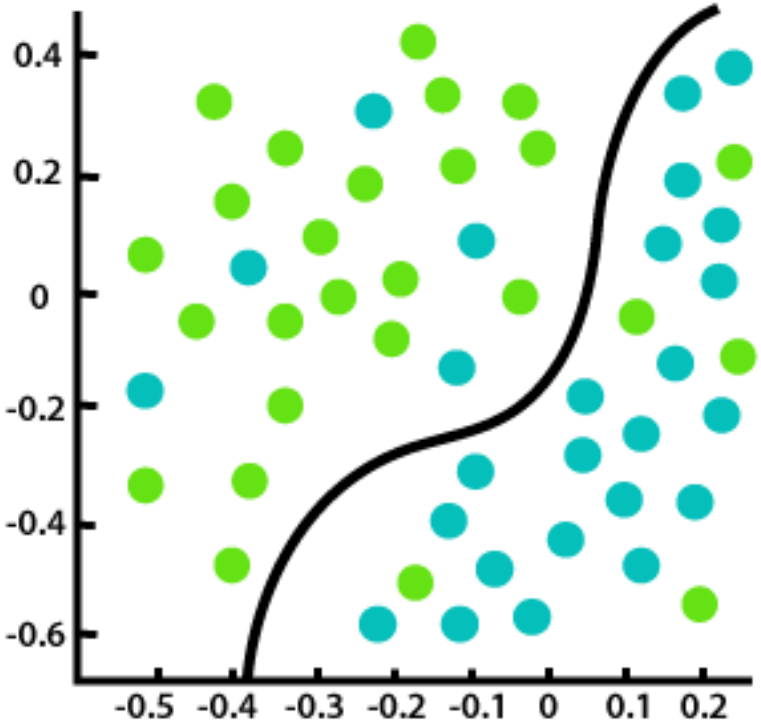


MACHINE LEARNING → 4 OBJETIVOS

NEURAL NETWORKS (MLP) → 2 OBJETIVOS



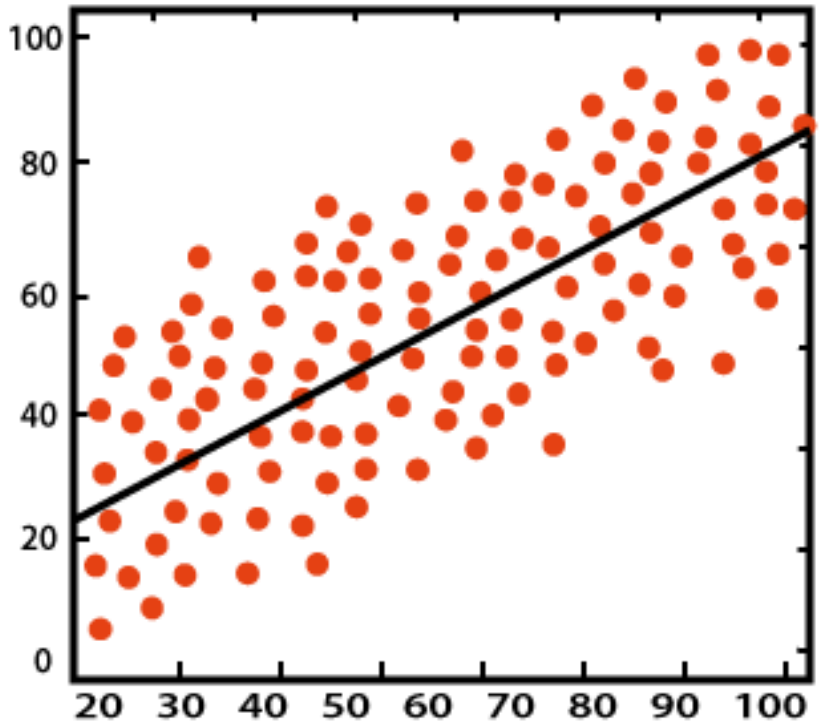
CLASSIFICAÇÃO VERSUS REGRESSÃO



Classification

Dados com labels discretos

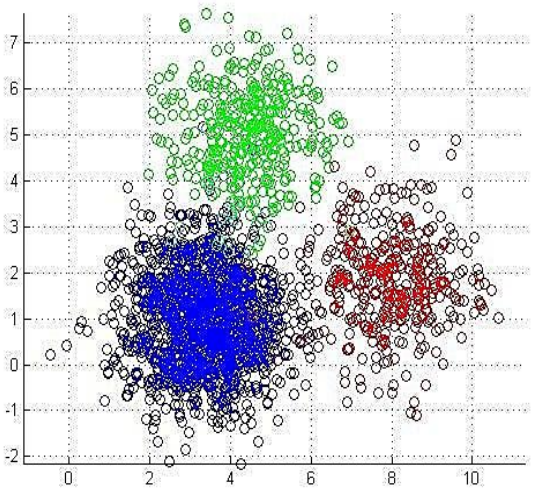
APP nprtool



Regression

Dados com labels contínuos

APP nrtool



Clusterization

Dados sem labels

ENTENDENDO O PROBLEMA

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

12 ENTRADAS

1 SAÍDA

3929 PADRÕES

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Amostra	Origem do	LL	IP	2"	1"	3/8"	# 4	# 10	# 40	# 200	Hot	Dmáx	Exp.	UTM-Este/X	UTM-Norte/Y	CBR/ISC
2	2896	BR-402/CE		NP	100	100	88	86	83	56	18	0,6	1,97	0	276505,152	9669694,607	23
3	4684	BR-116	NP	NP	100	91	73	66	58	47	29	1	2,005	0,1	498032,9802	9153085,231	15
4	497	Cariri (CE-	21	8	100	78	63	47	38	32	20	1	2,022	0	344292,2464	9205343,269	75
5	3043	BR-402/CE	NP	NP	100	100	100	100	100	91	36	1,3	1,885	0,1	407958,415	9627796,941	9
6	386	CE-292 (Tr	NL	NP	100	100	99	98	96	73	28	4,1	1,836	0,1	474676	9201105	14
7	653	Cariri (CE-	NL	NP	100	100	100	100	100	81	13	4,1	1832	0,2	469111,5149	9193580,023	14
8	690	Cariri (CE-	NL	NP	100	97	88	87	86	61	10	4,3	1898	0	480317,7879	9198693,996	14
9	656	Cariri (CE-	NL	NP	100	100	100	100	100	79	15	4,3	1789	0,2	470308,775	9193539,459	14
10	5296	BR-020	NP	NP	100	100	75	60	55	34	21	4,3	2,045	0	489179,189	9542186,77	60
11	378	CE-292 (Tr	NL	NP	100	100	100	100	99	79	6	4,4	1,787	0,2	476999,83	9199329,43	16
12	4165	BR-304	NP	NP	100	100	100	100	100	85	7	4,4	1,74	0	638888,159	9494795,565	19
13	4170	BR-304	NP	NP	100	100	100	100	100	87	15	4,5	1,781	0,1	638701,388	9494867,094	13

→ *nftool*

QUAL É A CLASSE DE PROBLEMA?

- Claramente não é um problema de classificação, até porque a variável a ser modelada é contínua e não discreta.
- Em problemas com essa característica nos dados o que se deseja é ou prever um valor e no Matlab essa classe de problema é chamada de **Regression**.
- Existe um APP para criar e treinar redes neurais com essa finalidade. Nosso arquivo chama-se solos_dnit.xlsx e deve ser colocado numa mesma pasta do scrip gerado no matlab.

CARREGANDO E PRÉ-PROCESSANDO OS DADOS

- Adote as boas práticas em qualquer modelagem e siga os mesmos procedimentos de pré-processamento adotados na classe de problema Classificação, mas com uma pequena diferença, qual seja: NÃO PRECISA (DEVEM) NORMALIZAR AS SAÍDAS, APENAS AS ENTRADAS
- A diferença fundamental entre classificar e estimar é que:
  - **Para classificação** usa-se na camada de saída a **função de ativação softmax** na camada de saída, que calcula a probabilidade da saída ser de uma determinada classe e optar por ela (0 ou 1, por exemplo;
  - **Para a previsão**, estimação (regressão) usa-se a **função de ativação identidade (linear)** na camada de saída, que permite que não altera o valor de saída, permitindo que ele assuma qualquer valor contínuo, por exemplo, os valores de CBR, ao normaliza-los a diferença entre eles seria muito pequena, dificultando o aprendizado da rede
- Sugere-se inclusive fazer uso do mesmo código anterior, com a diferença de não normalizar as saídas e adaptar ao nome e range do novo conjunto de dados.

PROMPT

“possuo um código em matlab e preciso altera-lo para que os dados da coluna Q do arquivo dados_Dnit.xlsx não sejam normalizados. O código é o seguinte:

```
% Carregar o arquivo excel
dataset = xlsread('dados_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
dataset = (dataset - min(dataset(:))) / (max(dataset(:)) - min(dataset(:)));
% 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
entradastreinamento = treinamento(:, 1:end-1); % Colunas E até P
saidatreinamento = treinamento(:, end); % Coluna Q
entradasteste = teste(:, 1:end-1); % Colunas E até P
saidateste = teste(:, end); % Coluna Q
```

“

A linha modificada é a seguinte:

```
colsToNormalize(colsToNormalize == size(dataset, 2)) = [];
colsToNormalize(colsToNormalize == 17) = [];
dataset(:, colsToNormalize) = (dataset(:, colsToNormalize) - min(dataset(:,
colsToNormalize))) ./ (max(dataset(:, colsToNormalize)) - min(dataset(:, colsToNormalize)));
```

Aqui, estamos removendo a coluna Q da lista de colunas a serem normalizadas. Isso garante que todas as outras colunas sejam normalizadas usando a função mini-max entre 0 e 1.

ALTERAÇÕES NO CÓDIGO

```
% Carregar o arquivo excel
dataset = xlsread('dadosexcel.xlsx', 1, 'B2:H1239);

% Criar uma ordem randômica das 1238 linhas
randomOrder = randperm(size(dataset, 1));
dataset = dataset(randomOrder, :);

% Normalizar todos os valores usando a função mini-max entre 0 e 1
dataset = (dataset - min(dataset(:))) / (max(dataset(:)) - min(dataset(:)));

% 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
entradastreinamento = treinamento(:, 1:end-1); % Colunas B até G
saidatreinamento = treinamento(:, end); % Coluna H

entradasteste = teste(:, 1:end-1); % Colunas B até G
saidateste = teste(:, end); % Coluna H
```

```
% 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 os valores usando a função mini-max entre 0 e 1, exceto da última coluna
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
entradastreinamento = treinamento(:, 1:end-1); % Colunas E até P
saidatreinamento = treinamento(:, end); % Coluna Q

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

```
segundo.m *
...r o arquivo excel
...set = 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(:

% 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
...adastreinamento = treinamento(:, 1:end-1); % Colunas
...inamento = treinamento(:, end); % Coluna Q
```

Run

Current Folder

Name

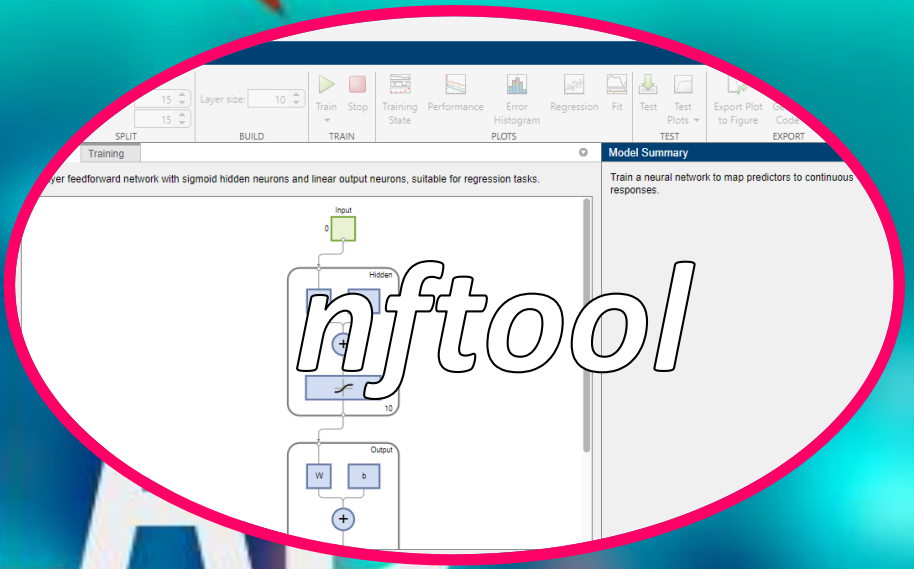
segundo.m

solos_dnit.xlsx

Name	Value
colsToNormalize	1x12 double
dataset	3929x13 double
entradasteste	1179x12 double
entradastreinamento	2750x12 double
numRows	3929
numTrain	2750
randomOrder	1x3929 double
saidateste	1179x1 double
saidatreinamento	2750x1 double
teste	1179x13 double
...reinamento	2750x13 doub...

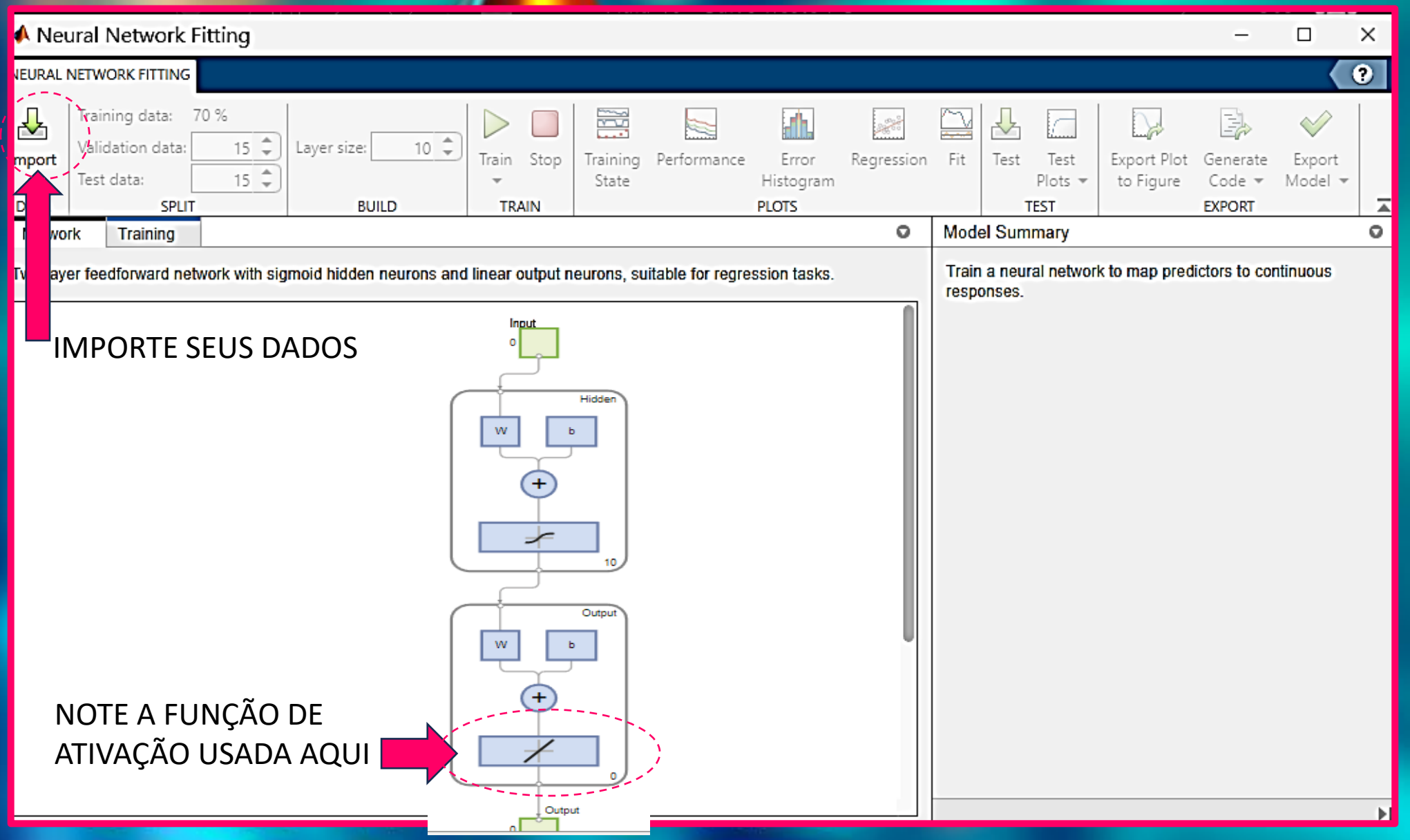
Command Window

```
>> nftool
```

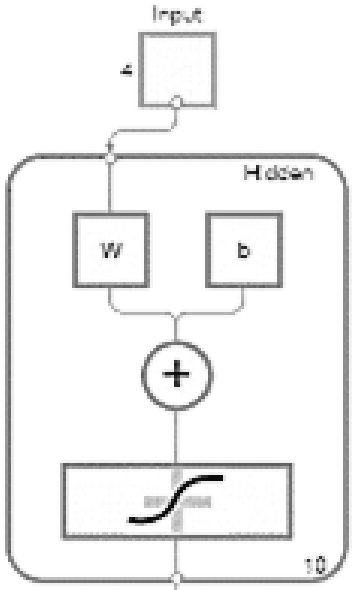


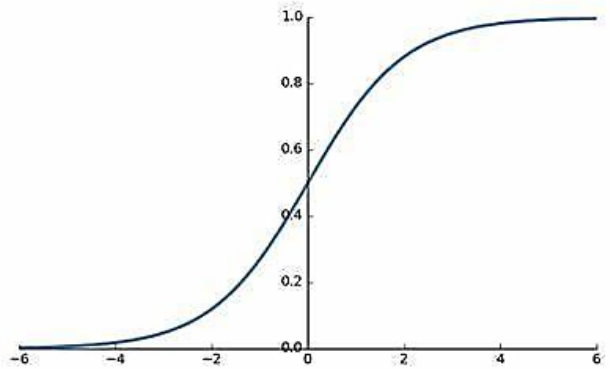
MATLAB

NEURAL NETWORK FITTING

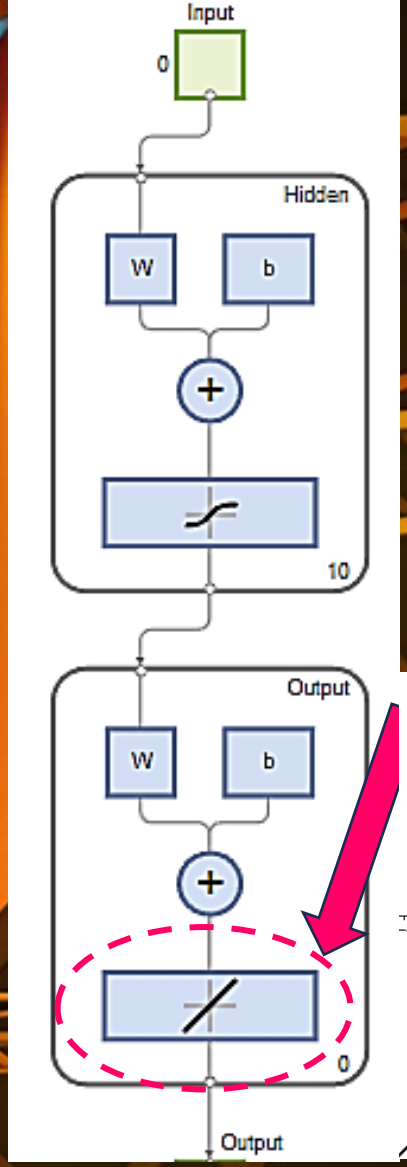


CLASSIFICAÇÃO



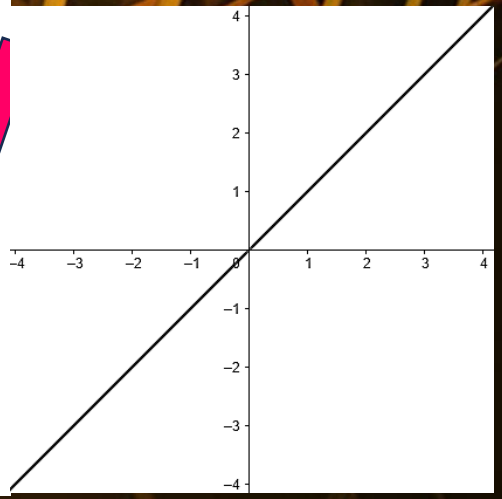


A função softmax é uma função de ativação que é comumente usada em redes neurais para classificação multiclasse . Ela converte um vetor de números em um vetor de probabilidades, onde a soma dos elementos do vetor resultante é igual a 1 .



A função linear é uma função de ativação simples que não altera a saída de um neurônio. Ela é geralmente utilizada nas camadas de saída em redes neurais de regressão.

REGRESSÃO



MATLAB

DEFININDO A ARQUITETURA DA REDE

O dataset nos indica que a rede deve ter 1 camada de entrada com 12 neurônios, uma camada intermediária e uma camada de saída com 1 neurônio.

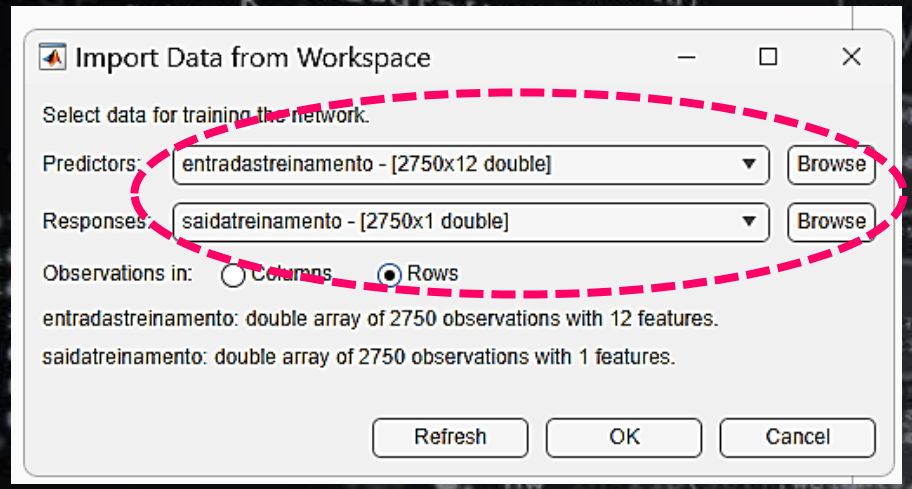
Sugere-se desenvolver uma árvore de modelagem para tentar encontrar a topologia mais adequada, mas para fins didáticos, neste exemplo usaremos as sugestões abaixo para determinar o número de neurônios da camada intermediária:

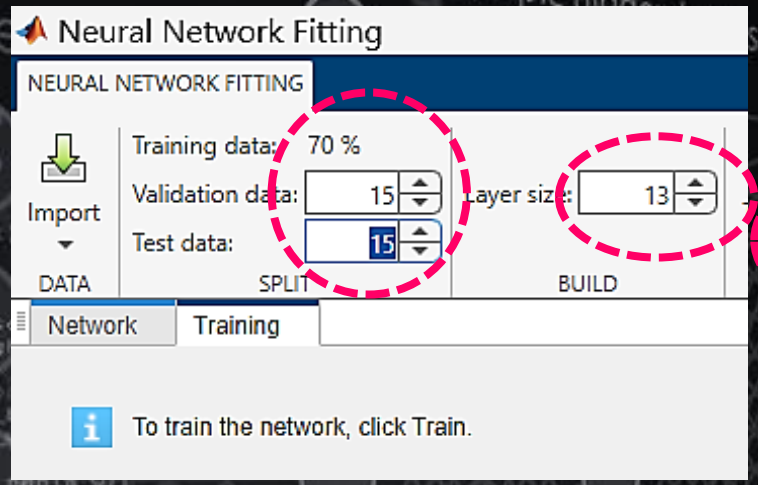
Segundo HETCH & NIELSEN, 1989 → A única camada intermediária deve ter 2i+1 neurônios → (**2. 12)+1=25 neurônios;**

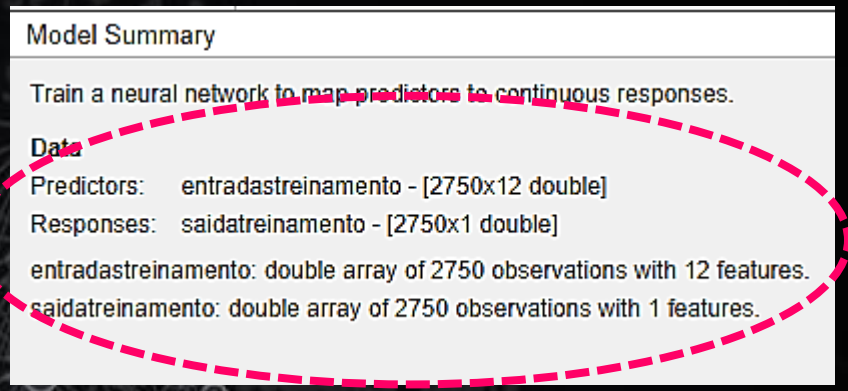
Já conforme LIPPMANN, 1987 → MLP de uma única camada devem ter s.(i+1) neurônios → **1(12+1)=13 neurônios**;

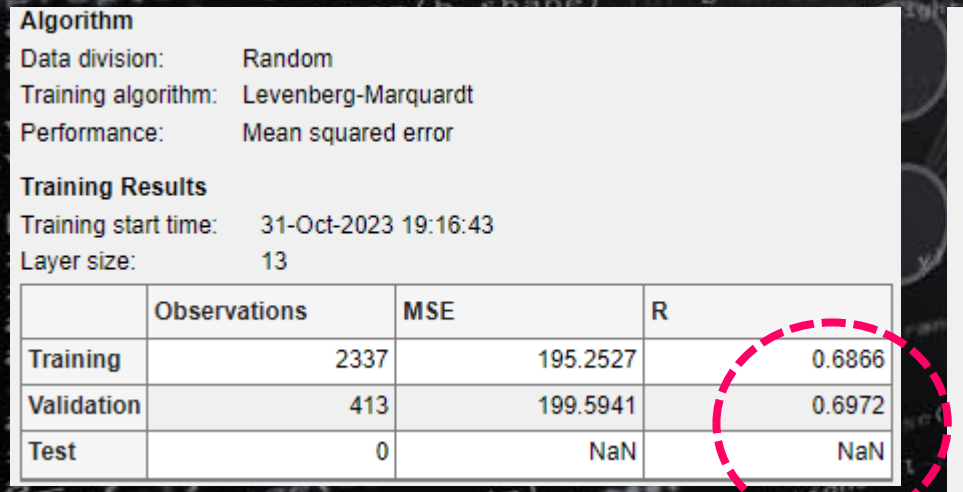
Para começar usaremos a sugestão de LIPPMANN e tentaremos a **Arquitetura→ 12:13:1** Caso o modelo não fique bom sugere-se acatar a outra sugestão com a seguinte **Arquitetura→ 12:25:1**

REDE 12:13:1

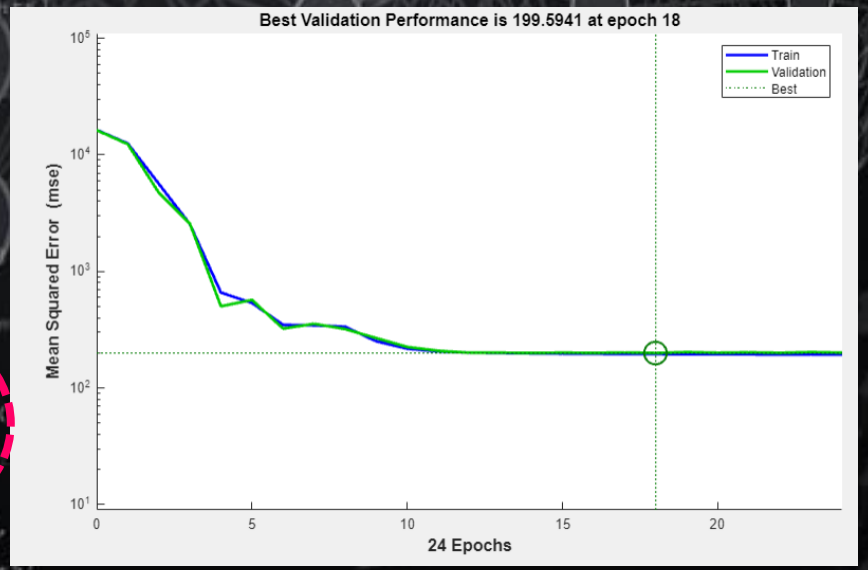




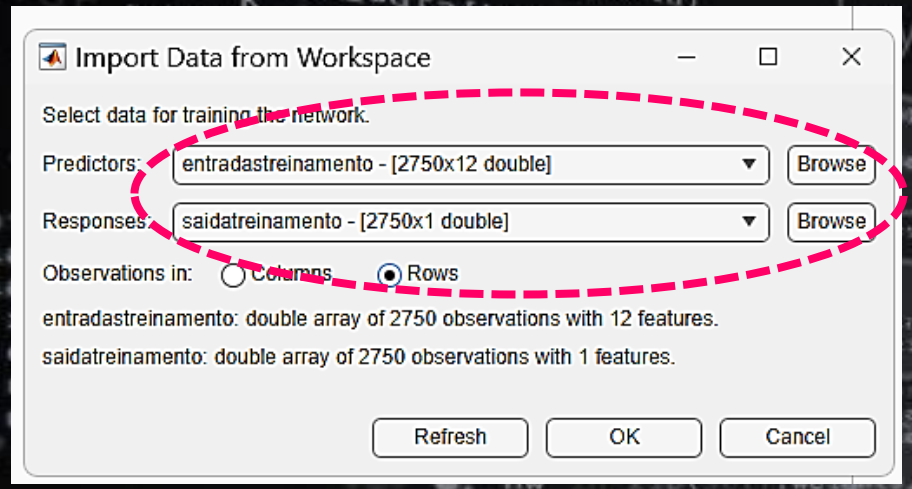


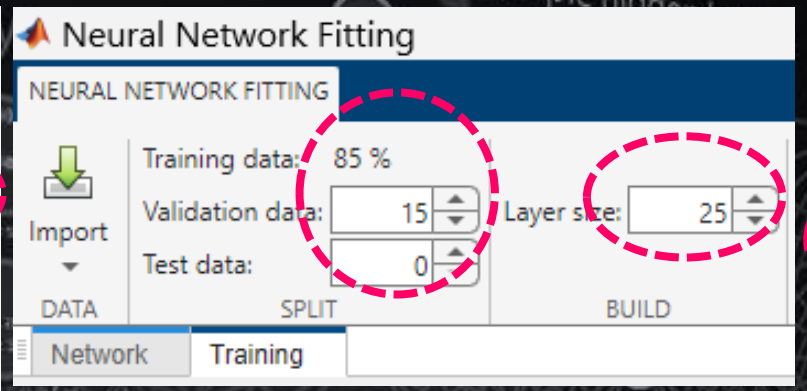


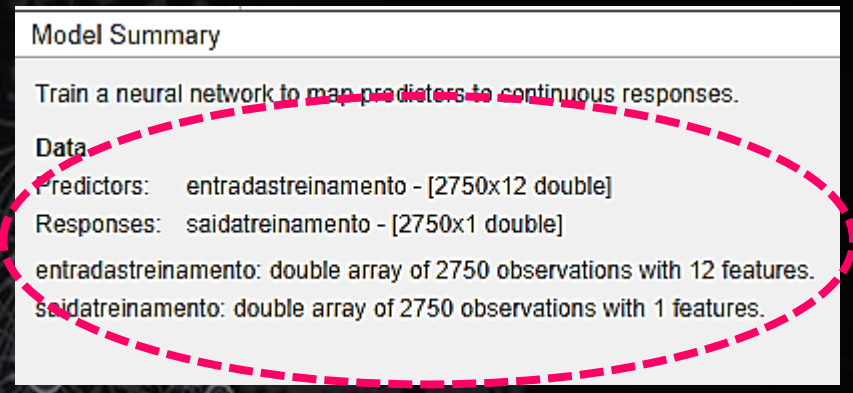
	Observations	MSE	R
Training	2337	195.2527	0.6866
Validation	413	199.5941	0.6972
Test	0	NaN	NaN



REDE 12:25:1







O que significa isso?
Significa que este problema é bem mais complexo que o anterior e que a busca de uma arquitetura mais adequada à generalização/previsão/estimação/ regressão é fundamental e aquela árvore de modelagem terá de ser realizada.

É um trabalho que demanda tempo e esforço do modelador, e não apenas variar o número de neurônios da única camada, alterar também os algoritmos, as funções de ativação, a taxa de aprendizagem, momento, critério de parada e outros hiperparâmetros e se nada disso melhorar, repetir tudo criando uma segunda camada e alterando o número de neurônios nela também..o ceú é o limite....

Algorithm

Data division: Random
Training algorithm: Levenberg-Marquardt
Performance: Mean squared error

Training Results

Training start time: 31-Oct-2023 19:21:13
Layer size: 25

	Observations	MSE	R
Training	2337	124.1128	0.8085
Validation	413	192.9434	0.7699
Test	0	NaN	NaN

COMPLEXIDADE DO FENÔMENO

1. Solos não são um material, são **misturas complexas**, encontrar as variáveis explicativas de seu comportamento mecânico é fundamental;
2. Sugere-se que além de ensaios geotécnicos, se **usem variáveis biofísicas, por exemplo, geologia e geomorfologia, das rochas que lhes deram origem (no caso de solos residuais), fitofisionomia, temperatura, indicadores meteorológicos, altitude, relevo e outras**;
3. Outro agravante é a **espacialidade das amostras**, criar um modelo que represente o comportamento dos solos de uma determinada região com uma área geográfica ampla como o estado do Ceará, requer que o dataset seja **representativo e que haja equilíbrio de classes** e, às vezes isso não é possível, devido a uma série de fatores como disponibilidade de recursos financeiros e tempo para um grande esforço amostral e um extenso programa experimental, incompatível com os recursos financeiros e de tempo disponíveis para pesquisa.

AVALIE A QUALIDADE DE SEU MODELO

1. Após encontrar o combo: algoritmo/arquitetura/hiperparâmetros mais adequado ao seu modelo, não esqueça de carregar aqueles 30%, esquecidos até agora, que correspondem ao conjunto de teste, dados esses que a rede nunca “viu”, nunca teve acesso, não foram usados em nenhum momento durante o processo de treinamento do modelo neural e,
2. Com os pesos do modelo agora já calibrados, teste a rede, fazendo com que ela estime/preveja as respostas (saídasteste) para as entradas apresentadas (entradasteste);
3. Baseado nos acertos e erros para o conjunto de teste e adotando um indicador que qualidade que o modelador escolher, por exemplo, RMS, apresente o erro médio quadrático da rede treinada ao estimar as respostas para o conjunto de entradas, com os quais nunca teve contato;
4. Só aí vai poder dizer se seu modelo ficou bom mesmo ou não.

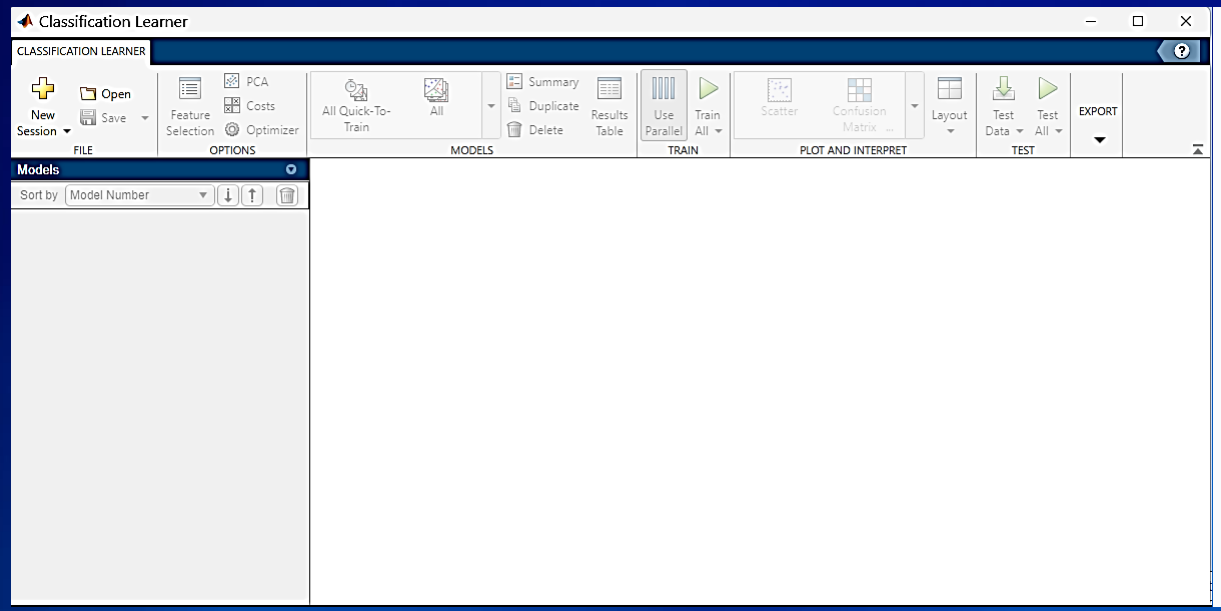
ANALISE A COMPLEXIDADE DO FENÔMENO

Após levar a cabo uma árvore de modelagem, cujo tamanho e esforço vai depender de diversos fatores, por exemplo:

1. Complexidade do problema;
2. Escolha das variáveis de entrada (independentes) que melhor representem o fenômeno investigado;
3. Tamanho, respresentatvidade (inclusive espaço/tempotal) e equilíbrio do dataset usado;
4. Escolha dos melhores: algoritmo, arquitetura e hiperparâmetros. E que se entenda melhores aqui como MAIS ADEQUADOS ao seu objetivo de modelagem

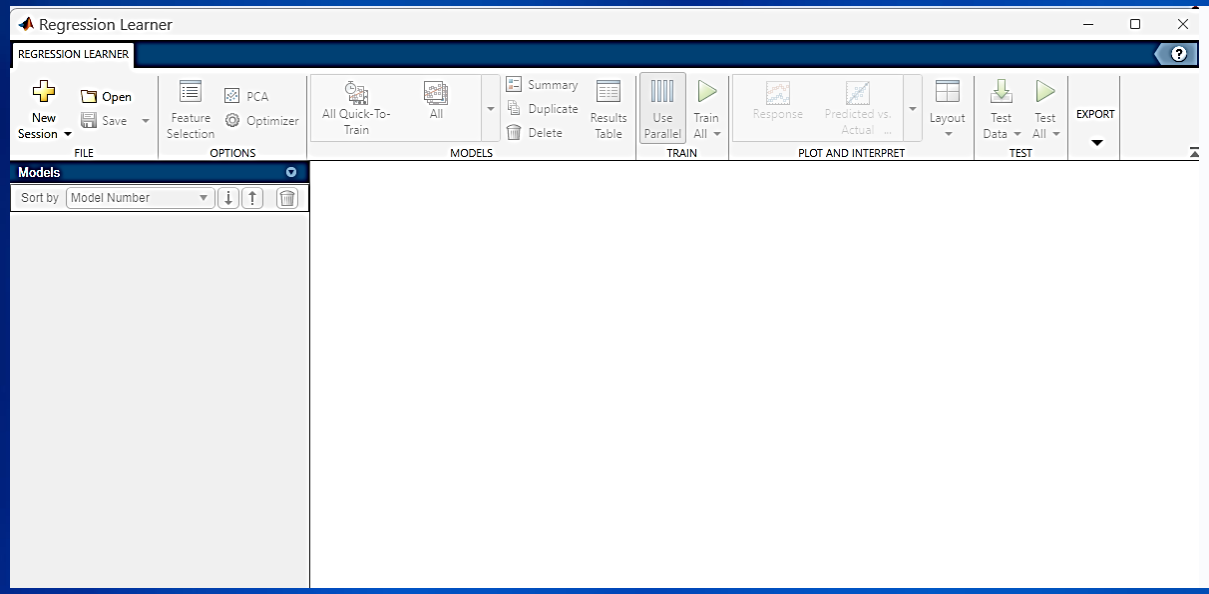
Agora, se mesmo depois de todo esse esforço, mesmo assim, constato empiricamente que Redes MLP clássicas não conseguem modelar adequadamente os dados dos quais disponho? Desisto de modelar? Claro que não! Apoie-se na literatura e tente outra técnica de modelagem ora!

Classification Learner



- Árvores de decisão: adequado para conjuntos de dados com muitas variáveis e classes.
- Florestas aleatórias: adequado para conjuntos de dados com muitas variáveis e classes.
- SVM (Máquinas de Vetores de Suporte): adequado para conjuntos de dados com muitas variáveis e classes.
- KNN (K-Nearest Neighbors): adequado para conjuntos de dados com poucas variáveis e classes.
- Redes neurais: adequado para conjuntos de dados com muitas variáveis e classes.
- Análise discriminante: adequado para conjuntos de dados com muitas variáveis e classes.
- Regressão logística: adequado para conjuntos de dados com duas classes.
- Naive Bayes: adequado para conjuntos de dados com muitas variáveis e classes.

Regression Learner



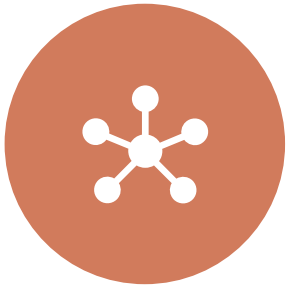
- Regressão linear: adequado para conjuntos de dados com uma única variável independente.
- Árvores de regressão: adequado para conjuntos de dados com muitas variáveis independentes.
- SVM (Máquinas de Vetores de Suporte): adequado para conjuntos de dados com muitas variáveis independentes.
- Regressão por processo gaussiano: adequado para conjuntos de dados com muitas variáveis independentes.
- Modelos de aproximação do kernel: adequado para conjuntos de dados com muitas variáveis independentes.
- Conjuntos de árvores de regressão: adequado para conjuntos de dados com muitas variáveis independentes.
- Redes neurais: adequado para conjuntos de dados com muitas variáveis independentes.

DICA IMPORTANTE:

“Se a essa altura do curso você ainda se mantiver com pulgas, imagino que já estarão, não apenas atrás da orelha. Neste caso, a solução não deve ser modelagem e sim algo como: NexGard, Simparic e outros.”

PONTOS CHAVE

O que é a classe de problemas de regressão



As diferenças entre redes para classificação e para regressão

O APP nrtool, suas características e uso para previsão

Criar, treinar e analisar os resultados de um modelo neural para estimar o CBR de solos no estado do Ceará


A SEGUIR CENAS DO
ÚLTIMO
CAPÍTULO
CRIAÇÃO DE MODELOS NEURAIS A PARTIR DE CÓDIGOS NO MATLAB