**Práctica de Redes Neuronales**

*El objetivo de esta práctica es trabajar con Redes Neuronales como modelo de clasificación empleando el entorno de simulación Matlab. Se emplearán datos bidimensionales para que puedan visualizarse fácilmente y se ilustre claramente el concepto de frontera de decisión.*

*La práctica se divide en varias etapas que habrá que programar en un script en Matlab comentado e identado. El script base es* clasificador\_neuronal.m

***1. Carga del fichero de datos.***

*El primer paso es cargar el fichero con los datos . Es el fichero que se os ha facilitado con nombre* datos\_train\_problem1.mat

*Funciones: load.*

**%============================================================**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%CARGA DE DATOS DE ENTRENAMIENTO %%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load datos\_train\_problem2;

% xtrain contiene los patrones de entrenamiento (dimensión 2)

% dtrain la clase del patrón (0/1 clasifici´on binaria)

**%============================================================**

# 2. Visualización de los datos

*En este apartado se debe realizar el script* visualizar\_datos\_2D(x,d)*que permita visualizar los datos* x *(datos con dos características). El vector* d *contiene las etiquetas de clase cada uno de los datos. Mostrar los datos de la clase 0 en rojo y los de la clase 1 en verde.*

*Funciones: plot, find.*

**%============================================================**

function []= visualizar\_datos\_2D(x,d)

% Script para visualizar los datos bidimensionales de la matriz x

% Mostrar los datos de la clase 0 en rojo y los de la clase 1 en verde.

posClass0 = find(d == 0);

class0 = x(:,posClass0);

posClass1 = find(d == 1);

class1 = x(:,posClass1);

plot(class0(1,:),class0(2,:),'rx');

hold on;

plot(class1(1,:), class1(2,:),'gx');

**%============================================================**

**%============================================================**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%VISUALIZACION DE DATOS %%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

visualizar\_datos\_2D(xtrain, dtrain) %% visualizar datos 2-D

**%============================================================**

# 3. Creación de la Red Neuronal

*Para esta práctica utilizaremos la toolbox de neuronales que viene incorporado en la instalación de MATLAB (Neural Network Toolbox). Se empleará una red con aquitectura MLP.*

*Funciones: newff.*

**%============================================================**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% GENERACIÓN DE LA NN %%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Necesitamos una red neuronal con dos neuronas de entrada y una de salida.

% Vamos a emplear un MLP x-10-1: 2 neuronas (nodos) de entrada, 10 neuronas en la capa oculta (logsig) y una neurona de salida (logsig).

% Y las funciones de activación de cada capa ({‘logsig’,’logsig’}

% Inicializamos la red net0

net0=newff([minmax(xtrain)],[10,1],{'logsig','logsig'},'trainlm');

**%============================================================**

1. ***Visualizar la frontera de decisión de la red y los datos de entrenamiento*** *Veremos la calidad de la clasificación de la red sin entrenar, es decir, inicializada con pesos aleatorios.*

**%============================================================**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% VISUALIZAR LA FRONTERA DE DECISION DE LA RED %%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Veremos la calidad de la clasificación de la red sin entrenar

vis\_fronteras(xtrain,dtrain,net0)

hold on;

visualizar\_datos\_2D(xtrain, dtrain)

title('Red no entrenada + datos train')

**%============================================================**

1. ***Evaluación de la Red Neuronal.***

*La evaluación se hará visualmente (fronteras de decisión y datos de test) y cuantitativamente con las tasas de error.*

*Funciones: sim, find, xor, length, Plot*

*Determinar los errores de clasificación para* datos\_test\_problem1.mat *y el número de ciclos de entrenamiento de la red neuronal.*

  

*E* ............... *E*0  ............... *E*1  ...............

**%============================================================**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% ENTRENAMIENTO DE LA NN %%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Numero de ciclos de entrenamiento

net0.trainParam.epochs=20

% Entrenamos el MLP (empleamos los parámetros por defecto de la función)

net=train(net0,xtrain,dtrain);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% EVALUACIÓN DE LA NN CON LOS DATOS DE TEST %%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load datos\_test\_problem2;

% xtest contiene los patrones de test (dimensión 2)

% dtest la clase del patrón (0/1 clasifici´on binaria)

% Salidas de la red para los datos de test

dred=sim(net,xtest);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% VISUALIZAR LA FRONTERA DE DECISION DE LA RED %%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Veremos la calidad de la clasificación de la red sin entrenar

figure();

vis\_fronteras(xtest,dtest,net)

hold on;

visualizar\_datos\_2D(xtest, dtest)

title('Red Entrenada + datos test')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% TASA DE ERROR DE LA RED %%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Calculo de la Tasa de Error (global y por clase)

% Etiqueta asignada para los datos de test con un umbral de 0.5

dtest\_asig=dred>0.5;

[Error,E0,E1]= calculo\_error (dtest, dtest\_asig)

**%============================================================**

# 6. Clasificador Neuronal para Problema 2

*Obtener un clasificador neuronal para los datos del problema 2.*

*(*datos\_train\_problem2.mat y datos\_test\_problem2.mat*)*

*Determinar los errores de clasificación para* datos\_test\_problem2.mat*y el número de ciclos de entrenamiento de la red neuronal.*

  

*E* ............... *E*0  ............... *E*1  ...............

**%============================================================**

function [Error,E0,E1]= calculo\_error (dtest, dtest\_asig)

% Script que calcule el error en la clasificacion de los datos dtest

% Devuelve

% E: Error global

% E0: Error en la clase 0

% E1: Error en la clase 1

totalError = xor(dtest,dtest\_asig);

Error = length(totalError(totalError == 1))/length(dtest);

posClass0 = find(dtest == 0);

posClass1 = find(dtest == 1);

class0Error = xor(dtest(posClass0),dtest\_asig(posClass0));

E0 = length(class0Error(class0Error == 1))/length(posClass0);

class1Error = xor(dtest(posClass1),dtest\_asig(posClass1));

E1 = length(class1Error(class1Error == 1))/length(posClass1);

**%============================================================**