```
function [w_opt, E_hist, E_val_hist] = descensoGradiente_val(xtrain,w,Ytrain,xval,Yval
```

Modificar

```
E_hist = zeros(1,num_iter); % inicializar a 0's el vector E_hist para almacenar el erro
E_val_hist = zeros(1,num_iter); % inicializar a 0's el vector E_hist para almacenar el

w_opt = zeros(2,num_iter); % inicializa a 0's el vector donde guardaremos en cada itera
w_opto = zeros(2,num_iter); % inicializa a 0's el vector donde guardaremos en cada ite
eps = 1e-6; % 1e-5 envalidacio per example % elige un valor de epsilon para que el buc
```

No Modificar

```
refresh_rate = num_iter/10; % solo actualiza la figura  cada 10 pasos
if fig, figure, end % crea una nueva figura si se quieren los plots

% bucle de iteraciones
for i = 1:num_iter
    E_hist(i) = calculaError(xtrain,w,Ytrain);  % calcula el error cuadratico medio pa
    E_val_hist(i) = calculaError(xval,w,Yval);
```

Modificar

```
    w_opt(:,i) = w;  % guardamos el valor actual de los pesos (solo lo usaremos para e
    y = w'*xtrain;  % calcula el output o predicci□n de la nuestra red
    e = y - Ytrain;  % calcula el error RELATIVO entre predicci□n y valor real
    grad_E = mean(repmat(e,size(xtrain,1),1).* xtrain ,2);  % calcula el gradiente del
    w = w - alpha*grad_E;  % actualiza el vector de los pesos con los nuevos valores (

    w_opto(:,i) = w;
    y = w'*xval;  % calcula el output o predicci□n de la nuestra red
    e = y - Yval;  % calcula el error RELATIVO entre predicci□n y valor real
    grad_E_val = mean(repmat(e,size(xval,1),1).* xval ,2);  % calcula el gradiente del
    w = w - alpha*grad_E;  % actualiza el vector de los pesos con los nuevos valores (

    if norm(grad_E) < eps % salir del bucle si la norma del gradiente es menor que eps
        disp(['Parado en la iteracion: ' num2str(i)])
        break
    end

    if norm(grad_E_val) < eps % salir del bucle si la norma del gradiente es menor que
        disp(['Parado en la iteracion: ' num2str(i)])
        break
    end
```

No modificar

plots

```
    if fig && mod(i,refresh_rate)==0

        subplot(2,2,1)
```

```
        hold off
        plot(xval(2,:), Yval, 'ro', 'linestyle', 'none', 'markersize', 8)
        hold on
        set(gca, 'fontsize', 10)
        xlabel('x'), ylabel('Y')
        plot(xval(2,:),y, 'b--', 'linewidth', 1)
        subplot(2,2,2)
        hold on
        plot_2Dtimeseries(w_opto(1,1:i), w_opto(2,1:i),1:i)
        if i==num_iter
        plot(w_opto(1,i),w_opto(2,i),'k+')
        end
        grid on
        xlabel('w_0'), ylabel('w_1')
        set(gca, 'fontsize', 10)
        subplot(2,2,3)
        plot(E_val_hist(1:i),'linewidth',2)
        set(gca, 'fontsize', 10,'xscale','log')
        xlabel('iter de validacion [log]'), ylabel('error de validación')
        xlim([1 num_iter])
%         subplot(2,2,4)
%         plot(E_val_hist(1:i),'linewidth',2)
%         set(gca, 'fontsize', 10,'xscale','log')
%         xlabel('iter de aprendizaje [log]'), ylabel('error de validación')
%         xlim([1 num_iter])
        drawnow
    end
end
```