Code for boolean functions task

```
%% Task to find linearly separable boolean functions for different dimensions
% Written by: Axel Qvarnström
clear all
close all
clc
n = 2; % Number of dimensions, just change to obtain result for different dimensions.
nTrials = 10^4;
nEpochs = 20;
eta = 0.05; % Learning rate
counter = 0;

booleanInputs = dec2bin(0:2^n-1)' - '0';
booleanInputs(booleanInputs == 0) = -1;     % Getting all zeros to -1
usedBooleanFuns = [];                       % A list for appending all already
                                            % tried boolean functions
k = 0;     % Index variabel used to store used boolean functions (see bottom of code)
for trial = 1:nTrials
    % Sample boolean functions
    booleanOutputs = randi([0,1],2^n,1);
    booleanOutputs(booleanOutputs == 0) = -1;


    boolFunInUsedBool = false;
    for i = 1:size(usedBooleanFuns,2)
        % To check if a boolean function is already used
        if isequal(usedBooleanFuns(:,i), booleanOutputs)
            boolFunInUsedBool = true;
            break
        end
    end
    if boolFunInUsedBool == true    % if so the case, skip to next iteration
        continue;
    end

    % Initializing weights and threshold
    weights = randn(n,1);
    threshold = 0;

    for epoch = 1:nEpochs
        totalError = 0;
        for mu = 1:2^n
            % Calculating the local field by summing (w(j)*x(j,mu) -
            % threshold) over j
            localField = 0;
            for j = 1:length(weights)
                localField = localField + (weights(j) * booleanInputs(j,mu) - threshold);
            end
```

```
            % Computing the output and checks if its equal to target or not
            y = sign(localField);
            error = booleanOutputs(mu) - y;

            % updating the weights and threshold
            deltaW = eta*(booleanOutputs(mu) - y).*booleanInputs(:,mu);
            deltaThreshold = -eta*(booleanOutputs(mu) - y);
            weights = weights + deltaW;
            threshold = threshold + deltaThreshold;

            totalError = totalError + abs(error);   % Calculating the total error

        end

        if totalError == 0
            counter = counter + 1;  % append everytime a boolean fun
                                    % is linearly separable
            break
        end
    end

    k = k+1;    % Index variable to append for every new boolean fun
                % into list for used boolean functions
    % Adding the boolean output to used boolean functions
    usedBooleanFuns(:,k) = booleanOutputs;

end

% Creating a variable for the nr of linearly separable functions
% and printing the result
nrOfLinearlySeparableFun = counter;
fprintf('Number of linearly separable functions for n = %0.f dimensions is %0.f'...
,n,nrOfLinearlySeparableFun);
```