Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Industriales

Máster en Automática y Robótica

Applied Artificial Intelligence

Assignment 4.1: Classification Error

Josep María Barberá Civera (17048)

April 6, 2024

ASSIGNMENT 4.1

Classification Error

Main instructions for the task completion are presented here:

1. First complete the code in order to evaluate the classification errors (both training and test using a Bayesian classifier for an increasing number of training samples.
2. Draw the resulting figure.
3. And finally discuss your results.

1.1 Methodology

The solution for the code with gaps is presented in the following image (Fig. 1. The overall structure from the provided code has been changed in order to better visualise the flow of the program.

```
for i = 1:length(ns)
    nsi = ns(i);
    for j = 1:nt2a
        ind_rand = randperm(N);
        ind = ind_rand(1:nsi);
        while ( (length(find(p.class(:,ind) == 1)) < 3) || ...
                (length(find(p.class(:,ind) == 2)) < 3) )
            ind_rand = randperm(N);
            ind = ind_rand(1:nsi);
        end
        bayMdl = fitcnb( p.value(:,ind)', p.class(:,ind)' );
        bayclass_train = predict( bayMdl, p.value(:, ind)');
        bayclass_test = predict( bayMdl, t.value');
        error_train(j,i)=length(find(bayclass_train' ~= p.class(:, ind)));
        error_test(j,i) = length(find(bayclass_test' ~= t.class ));
    end
    error_train_m(i) = mean(error_train(:,i)) / nsi;
    error_test_m(i) = mean(error_test(:,i)) / Nt;
end
```

Figure 1: *Matlab code with the solutions for the exercise. This part of the code does iterative training with increasing number of samples.*

In the code, the training error is validated with the same number and value of data, while the test error is calculated with all validation data values. The evolution of both errors with increasing number of training data is included in the following section.

1.2 Discussion and Results

The following figure (Fig. 2) shows training and test errors versus the number of training data. It can be seen that the higher the number of data, the training error increases, while the test error (the one that really matters to us) decreases. In short, what can be seen is the increase in the generalisation capacity. In other words, for a small amount of data, we obtain almost zero error when evaluating the same data. As more data becomes available, the error increases precisely because of this generalisation.

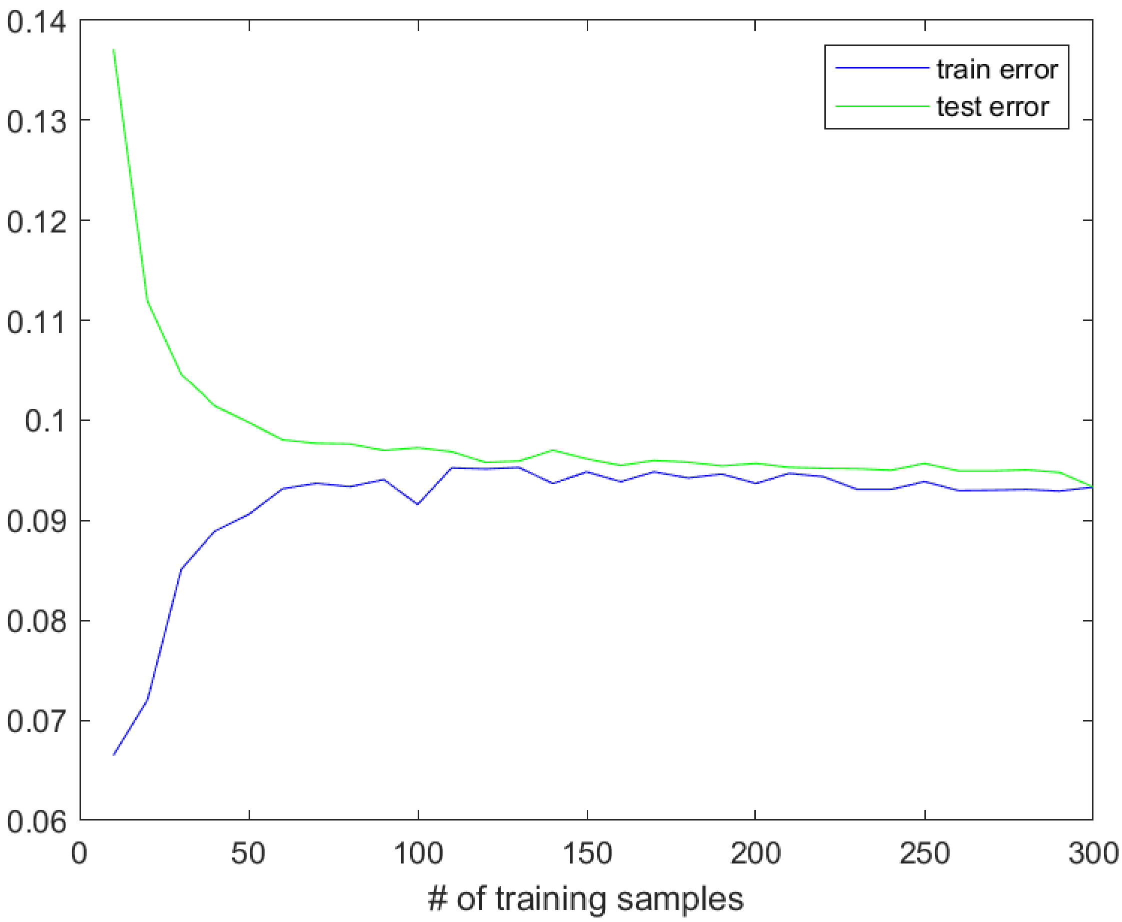

Figure 2: *Training and test error vs. the number of training samples. Both error tend to the Bayesian error which is intrinsic to the problem and cannot be minimised.*

1.3 Relevant Code

The code prepared for this assignment is shown in the next pages in a wider style.

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                            Master in Robotics
%                     Applied Artificial Intelligence
%
% Assinment 4.1:  Classification Error
% Student: Josep Barbera Civera
% ID: 17048
% Date: 06/04/2024
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% 0. Complete the code in order to evaluate the classification
% errors (both training and test) using a Bayesian classifier for
% an increasing number of training samples.
% 1. Draw the resulting figure.

load data_D2_C2.mat;

[D, N] = size(p.value);
[D, Nt] = size(t.value);

ns = 10:10:N;
nt2a = 400;
for i = 1:length(ns)
    nsi = ns(i);
    for j = 1:nt2a
        ind_rand = randperm(N);
        ind = ind_rand(1:nsi);
        while ( (length(find(p.class(:,ind) == 1)) < 3) || ...
                (length(find(p.class(:,ind) == 2)) < 3) )
            ind_rand = randperm(N);
            ind = ind_rand(1:nsi);
        end
        bayMdl = fitcnb( p.value(:,ind)', p.class(:,ind)' );
        bayclass_train = predict( bayMdl, p.value(:, ind)');
        bayclass_test = predict( bayMdl, t.value');
        error_train(j,i)=length(find(bayclass_train'~=p.class(:,ind)));
        error_test(j,i) = length(find(bayclass_test' ~= t.class ));
    end
    error_train_m(i) = mean(error_train(:,i)) / nsi;
    error_test_m(i) = mean(error_test(:,i)) / Nt;
end

plot(ns, error_train_m, 'b', ns, error_test_m, 'g');
legend('train error', 'test error');
xlabel('# of training samples');
saveas(gca, "trainin_and_test_error.png");
```