ECE 271A: Statistical Learning I

Homework 3

Shang-Wei Hung
A53267981

Problem 4

Posterior distribution for μ: $P_{\mu|T}(\mu|D_i)$=G(μ, μ_i, σ_i^2), where μ_i and σ_i^2 can be expressed as follow:

$$\mu_n = \frac{\sigma_0^2 \sum_i x_i + \mu_0 \sigma^2}{\sigma^2 + n\sigma_0^2} = \frac{n\sigma_0^2}{\sigma^2 + n\sigma_0^2}\ \mu_{ML}\ +\ \frac{\sigma^2}{\sigma^2 + n\sigma_0^2}\ \mu_0$$

$$\sigma_n^2 = \frac{\sigma^2\sigma_0^2}{\sigma^2 + n\sigma_0^2}$$

$$\sigma^2: variance\ of\ the\ dataset$$

$$\mu_0, \sigma_0^2: mean\ and\ variance\ given\ in\ the\ strategy, respectively$$

Gaussian function multiply Gaussian function is still Gaussian function. Thus,

$$P_{X|T}(x|D_n)\ = \text{G}(x, 0, \sigma^2) * \text{G}(x, \mu_n, \sigma_n^2) = \text{G}(x, \mu_n, \sigma^2 + \sigma_n^2)$$

Figure 1, Figure2, Figure 3, and Figure 4 shows the Bayes Decision Rule based on three different ways. (i) the predictive distribution (ii) the MAP estimate (iii) the ML estimate on different dataset from D1 to D4 with strategy 1(different μ_0)

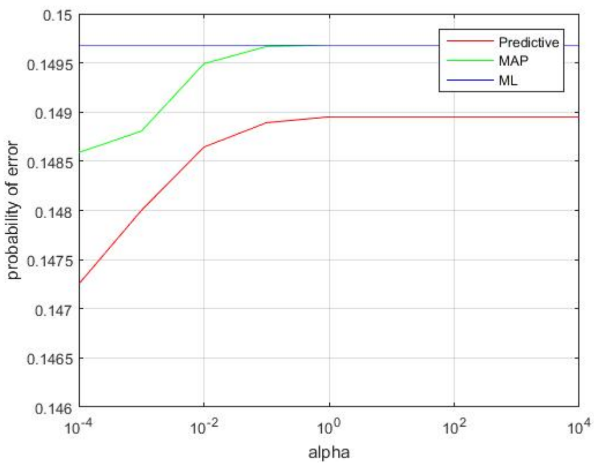

Figure 1: Probability of error vs. α
(***D1*** *Dataset* and *Strategy 1)*

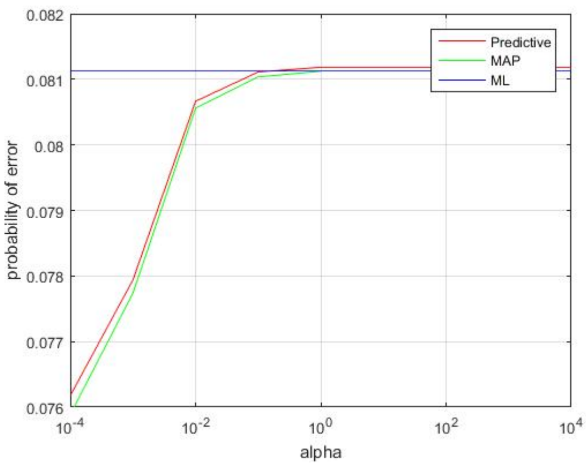

Figure 2: Probability of error vs. α
(***D2*** *Dataset* and *Strategy 1)*

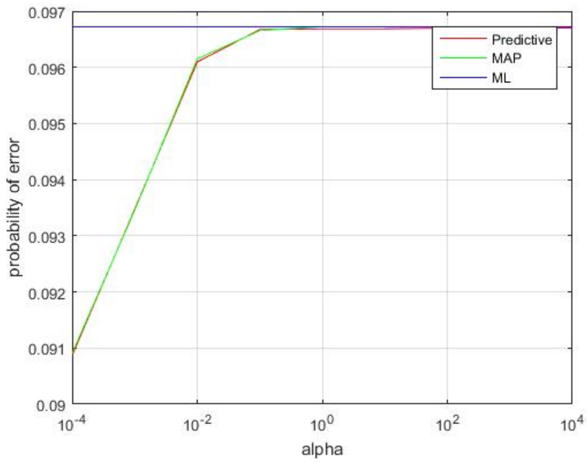

Figure 3: Probability of error vs. α
(***D3*** *Dataset* and *Strategy 1)*

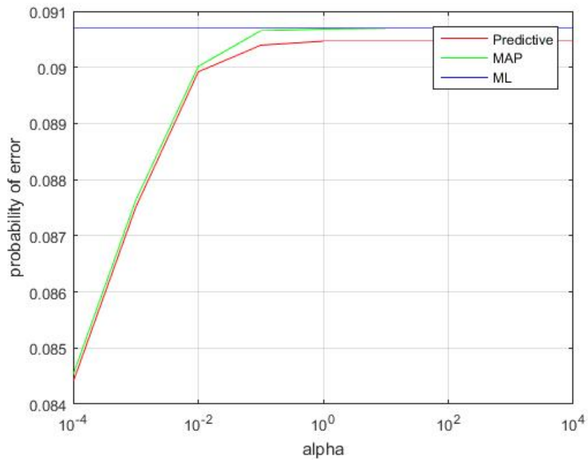

Figure 4: Probability of error vs. α
(***D4*** *Dataset* and *Strategy 1)*

As α increases, the four plots (D1,D2,D3,D4) of predictive solutions also increase, and converge to the ML solutions. The reason lies in the dominance of the prior. Given the assumed diagonal covariance matrix $(\Sigma_0)_{ii} = \alpha w_i$, when α increases, the prior become unreliable.
On the other hand, the prior helps a lot in the classification when α is small. Thus, since the predictive solution combined the prior belief (strategy 1) with the evidence provided in the data, the probability of error of these four datasets decrease as α decreases due to the precise prior.

For ML solution, since the mean and covariance are directly derived from the dataset without using parameter α, the probability of error will not chance as α increases. Thus, the lines of probability of error are all horizontal lines for these four datasets(D1,D2,D3,D4).

$P_{X|T}(x|D_i)$= $P_{X|\mu}(x|\mu_{MAP})$,where μ_{MAP}=arg $max_{\mu}\, P_{\mu|T}(\mu|D_i)$
The covariance of the MAP solution is as same as the ML solution, while the mean of the MAP solution is derived as same as the predictive solution. Thus, MAP solution use the prior information to some extent. Hence, as the parameter α increase, the probability of error also increase for these four datasets(D1,D2,D3,D4).

The classfication errors with differenet solutions are sorted as below:

Error of Predictive Solution < Error of MAP Solution < Error of ML Solution

In general, the predictive solution and the MAP solution converge to the ML solution as α increase. The reason is that the predictive solution use all the information, and the MAP solution use prior to some extent. Since the prior is precise in strategy 1, the probability of error of Predictive Solution and MAP Solution are smaller than the error of ML solution.

The classfication errors with differenet datasets are sorted as below:

Error of D2 < Error of D4 < Error of D3 <Error of D1

Idealy, the probabiltiy of classification error should decrease as the size of training dataset becomes larger. But in this case, all datasets are still not sufficient large, it's therefore reasonale to have this sorted result of probability of error.

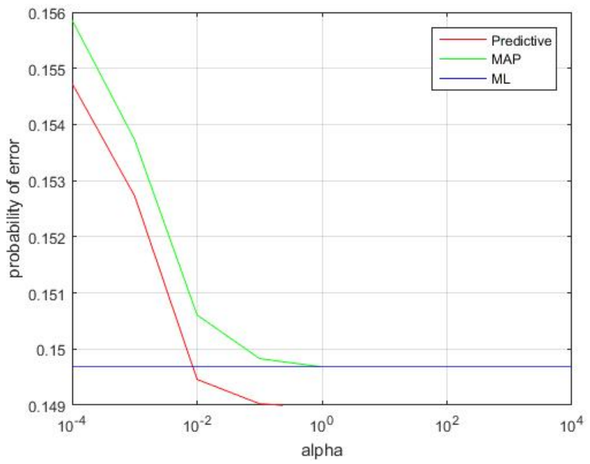

Figure 5: Probability of error vs. α
(***D1*** *Dataset* and *Strategy 2)*

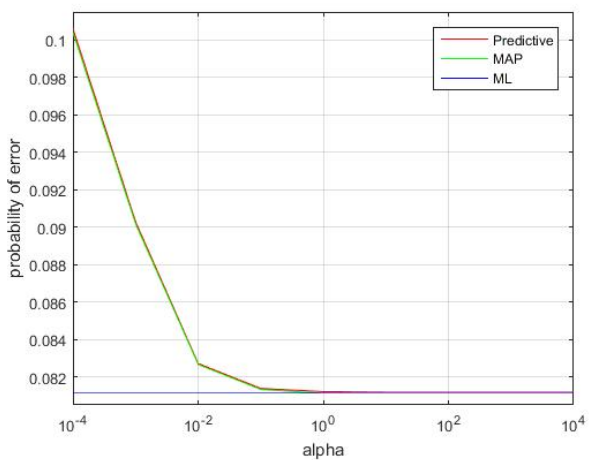

Figure 6: Probability of error vs. α
(***D2*** *Dataset* and *Strategy 2)*

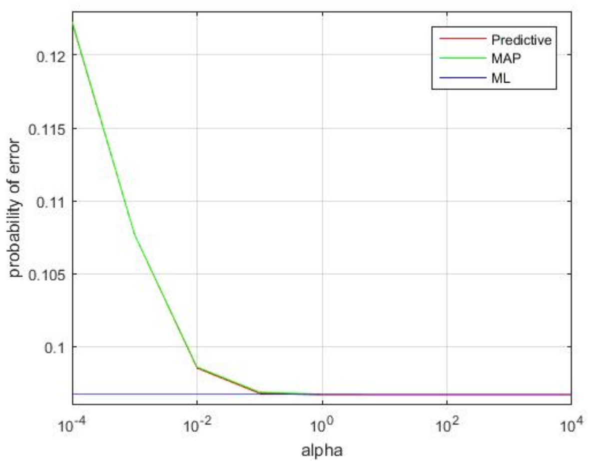

Figure 7: Probability of error vs. α
(***D3*** *Dataset* and *Strategy 2)*

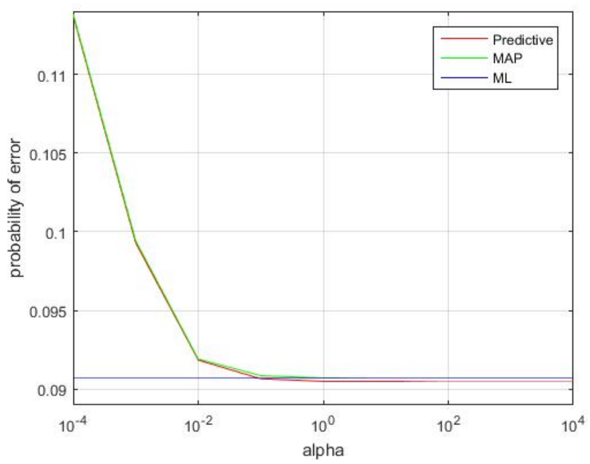

Figure 8: Probability of error vs. α
(***D4*** *Dataset* and *Strategy 2)*

Now, strategy 1 is replaced by strategy 2.

As α increases, the four plots (D1,D2,D3,D4) of predictive solution decrease, and converge to the ML solutions. The reason is that the priors' means of two classes are identical. In order words, the prior is not precise. Therefore, as α increases (rely less on unprecise prior), the probability of error decreases for these four datasets(D1,D2,D3,D4).

For ML solution, the result is as same as the one using strategy 1 in previous question. Since the mean and covariance are directly derived from the dataset without using parameter α, the probability of error will stay unchanged as α increases.

For MAP solution, since the mean of the MAP solution is derived as same as the predictive solution, the MAP solution use the prior information to some extent. Hence, as the parameter α increases (rely less on unprecise prior), the probability of error decrease for these four datasets (D1,D2,D3,D4).

Generally, since the prior is not precise now, the solutions (Predictive solution and MAP solution) using prior will perform worse classification than one without using any prior information (ML solution).

Unlike strategy 1, strategy 2 provides unprecise prior. Thus, the predictive solution (use all the prior information)and the MAP solution (use the prior information to some extent) in general become worse than the ML solution.

Moreover, the sorted result of probability of error is

Error of D2 < Error of D4 < Error of D3 <Error of D1.

Since all the datasets' size are not sufficient large, the sorted probability of error cannot follow that the larger dataset will definitely increase the classification precision.

Code: (Main part + Function file)

```
close all
clear all
clc

load('TrainingSamplesDCT_subsets_8.mat')
```

```
load('Alpha.mat')
load('Prior_1.mat')
% Load Zig-Zag pattern
Zigzag=load('Zig-Zag Pattern.txt');
Zigzag=Zigzag+1;

%%
% Change dataset here
FG = D1_FG;
BG = D1_BG;
[row_f,col_f] = size(FG);
[row_b,col_b] = size(BG);

% Calculate Prior
Prior_FG=row_f/(row_f+row_b);
Prior_BG=row_b/(row_f+row_b);

% calculate for each column's mean
mu_fg=sum(FG)/row_f;
mu_bg=sum(BG)/row_b;

% calculate covariance
covar_fg=cov(FG);
covar_bg=cov(BG);

% Load Ground Truth
% Load Test sample
Img_ori=imread('cheetah.bmp');                    %Use zero padding for edge &
corner
Img = padarray(Img_ori,[7 7],'symmetric','pre');    %classified pixel:right
bottom
Img=im2double(Img);
Gt=imread('cheetah_mask.bmp');
Gt=im2double(Gt);

Errors_Bay=zeros(1,length(alpha));
Errors_MAP=zeros(1,length(alpha));
Errors_ML=zeros(1,length(alpha));
```

```
%% Predictive solution
for k=1:length(alpha)
    k
    % Generate data
    [row,col]=size(Img);
    A=zeros((row-7)*(col-7),64);
    index=1;
    for i=1:row-7
        for j=1:col-7
            Field=Img(i:i+7,j:j+7);
            DCT=dct2(Field);
            DCT_64(Zigzag)=DCT;  % turn 8*8 into 1*64 with Zigzag pattern
            A(index,:)=DCT_64;
            index=index+1;
        end
    end


    %calculate mu1 for both class
    Sigma_0_FG = diag(W0)*alpha(k);
    sig_FG = covar_fg/row_f;
    first_term_FG = Sigma_0_FG /(sig_FG + Sigma_0_FG) * mu_fg';
    second_term_FG = sig_FG /(sig_FG + Sigma_0_FG) * mu0_FG';
    mu_1_FG = first_term_FG + second_term_FG;
    mu_1_FG = mu_1_FG';

    Sigma_0_BG = diag(W0)*alpha(k);
    sig_BG = covar_bg/row_b;
    first_term_BG = Sigma_0_BG /(sig_BG + Sigma_0_BG) * mu_bg';
    second_term_BG = sig_BG /(sig_BG + Sigma_0_BG) * mu0_BG';
    mu_1_BG = first_term_BG + second_term_BG;
    mu_1_BG = mu_1_BG';

    %calculate var1 for both class
    var_1_FG = (sig_FG * Sigma_0_FG) /(sig_FG + Sigma_0_FG);
    var_1_BG = (sig_BG * Sigma_0_BG) /(sig_BG + Sigma_0_BG);
```

```
    % for X|T
    varXT_FG = covar_fg + var_1_FG;
    varXT_BG = covar_bg + var_1_BG;



    % Plug into bayesian decision rule
    alp_fg=log(((2*pi)^64)*det(varXT_FG))-2*log(Prior_FG);
    alp_bg=log(((2*pi)^64)*det(varXT_BG))-2*log(Prior_BG);

    g_fg=zeros((col-7)*(row-7),1);
    g_bg=zeros((col-7)*(row-7),1);
    temp_dxy_fg=zeros((col-7)*(row-7),1);
    temp_dxy_bg=zeros((col-7)*(row-7),1);

    predict=zeros(1,(col-7)*(row-7));
    for index=1:(col-7)*(row-7)
        temp_dxy_fg(index)=(A(index,:)-mu_1_FG) * (inv(varXT_FG)* (A(index,:)-
mu_1_FG)');
        temp_dxy_bg(index)=(A(index,:)-mu_1_BG) * (inv(varXT_BG)* (A(index,:)-
mu_1_BG)');
        g_fg(index)=1 / (1+ exp( temp_dxy_fg(index) - temp_dxy_bg(index) +
alp_fg - alp_bg));
        g_bg(index)=1 / (1+ exp( temp_dxy_bg(index) - temp_dxy_fg(index) +
alp_bg - alp_fg));

        if g_fg(index)>0.5
            predict(1,index)=1;
        end
    end

    predict_2d=reshape(predict,[col-7,row-7]);
    predict_2d=predict_2d';

    Errors_Bay(k)=ProbOfError(predict_2d,Gt,Prior_FG,Prior_BG);
    Errors_Bay(k)

%% MAP
    % Plug into bayesian decision rule
```

```
    alp_fg=log(((2*pi)^64)*det(covar_fg))-2*log(Prior_FG);
    alp_bg=log(((2*pi)^64)*det(covar_bg))-2*log(Prior_BG);

    g_fg=zeros((col-7)*(row-7),1);
    g_bg=zeros((col-7)*(row-7),1);
    temp_dxy_fg=zeros((col-7)*(row-7),1);
    temp_dxy_bg=zeros((col-7)*(row-7),1);

    predict=zeros(1,(col-7)*(row-7));
    for index=1:(col-7)*(row-7)
        temp_dxy_fg(index)=(A(index,:)-mu_1_FG) * (inv(covar_fg)* (A(index,:)-
mu_1_FG)');
        temp_dxy_bg(index)=(A(index,:)-mu_1_BG) * (inv(covar_bg)* (A(index,:)-
mu_1_BG)');
        g_fg(index)=1 / (1+ exp( temp_dxy_fg(index) - temp_dxy_bg(index) +
alp_fg - alp_bg));
        g_bg(index)=1 / (1+ exp( temp_dxy_bg(index) - temp_dxy_fg(index) +
alp_bg - alp_fg));

        if g_fg(index)>0.5
            predict(1,index)=1;
        end
    end

    predict_2d=reshape(predict,[col-7,row-7]);
    predict_2d=predict_2d';

    Errors_MAP(k)=ProbOfError(predict_2d,Gt,Prior_FG,Prior_BG);
    Errors_MAP(k)

%% ML
    % Plug into bayesian decision rule
    alp_fg=log(((2*pi)^64)*det(covar_fg))-2*log(Prior_FG);
    alp_bg=log(((2*pi)^64)*det(covar_bg))-2*log(Prior_BG);

    g_fg=zeros((col-7)*(row-7),1);
    g_bg=zeros((col-7)*(row-7),1);
    temp_dxy_fg=zeros((col-7)*(row-7),1);
```

```
    temp_dxy_bg=zeros((col-7)*(row-7),1);

    predict=zeros(1,(col-7)*(row-7));
    for index=1:(col-7)*(row-7)
        temp_dxy_fg(index)=(A(index,:)-mu_fg) * (inv(covar_fg)* (A(index,:)-
mu_fg)');
        temp_dxy_bg(index)=(A(index,:)-mu_bg) * (inv(covar_bg)* (A(index,:)-
mu_bg)');
        g_fg(index)=1 / (1+ exp( temp_dxy_fg(index) - temp_dxy_bg(index) +
alp_fg - alp_bg));
        g_bg(index)=1 / (1+ exp( temp_dxy_bg(index) - temp_dxy_fg(index) +
alp_bg - alp_fg));

        if g_fg(index)>0.5
            predict(1,index)=1;
        end

    end

    predict_2d=reshape(predict,[col-7,row-7]);
    predict_2d=predict_2d';

    Errors_ML(k)=ProbOfError(predict_2d,Gt,Prior_FG,Prior_BG);
    Errors_ML(k)
end




    figure;
    semilogx(alpha, Errors_Bay, '-r',alpha, Errors_MAP, '-g',alpha, Errors_ML,
'-b');
    grid
    ylim([0.1460 0.1500])
    legend('Predictive','MAP','ML')
    xlabel('alpha');
    ylabel('probability of error');
```

```matlab
function Prob_Error=ProbOfError(predict_2d,Gt,Prior_FG,Prior_BG)
    Errors_FG=0;
    Errors_BG=0;
    [row,col] = size(Gt);

    Gt_FG=0;
    Gt_BG=0;
    for i=1:row
        for j=1:col
            if Gt(i,j)==1
                Gt_FG=Gt_FG+1;
            end
            if Gt(i,j)==0
                Gt_BG=Gt_BG+1;
            end
        end
    end

    for i=1:row
        for j=1:col
            if Gt(i,j)==1 && predict_2d(i,j)==0 % FG pixels, misclassifcied as BG
                Errors_FG=Errors_FG+1;
            end
            if Gt(i,j)==0 && predict_2d(i,j)==1
                Errors_BG=Errors_BG+1;
            end
        end
    end

    Errors_FG_p=Errors_FG/Gt_FG;    %Type II (False Negative)
    Errors_BG_p=Errors_BG/Gt_BG;    %Type I  (False Positive)

    Prob_Error=Errors_FG_p*Prior_FG + Errors_BG_p*Prior_BG;

end
```