Homework Assignment 3

Quan Luo
Statistical Learning I - Fall 2021

November 14, 2021

1 Basic Deduction for High Dimension Case

For this high dimensional case, we have:

$$\begin{aligned} P_{\mu|T}(\mu|D) &\propto P_{T|\mu}(D|\mu)P_{\mu}(\mu) \propto \prod_{i=1}^{n} P(X_i|\mu)P_{\mu}(\mu) \\ &\propto \prod_{i=1}^{n} G(x_i, \mu, \sigma^2)G(\mu, \mu_0, \sigma_0^2) \\ &\propto exp\{-\frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu) - \frac{1}{2}(\mu-\mu_0)^T\Sigma_0^{-1}(\mu-\mu_0)\} \\ &\propto exp\{-\frac{1}{2}\mu^T(\Sigma_0^{-1}+n\Sigma^{-1})\mu + (\sum_{i=1}^{n} x_i^T\Sigma^{-1} + \mu_0^T\Sigma_0^{-1})\mu\} \end{aligned}$$

From above deduction and with further step calculation we can conclude that:

$$\mu_1 = \Sigma_0(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}(\frac{1}{n}\sum_{i=1}^{n} x_i) + \frac{1}{n}\Sigma(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}\mu_0$$

$$\Sigma_1 = \Sigma_0(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}\Sigma\frac{1}{n}$$

From that we know that for predictive distribution:

$$\mu_{pred} = \mu_1,\ \Sigma_{pred} = \Sigma_1 + \Sigma$$

According to the deduction above, we can easily experiment with the three cases to show the classification results namely by predictive distribution, MAP and ML (Codes are attached at last).

2 Experiments of Strategy 1

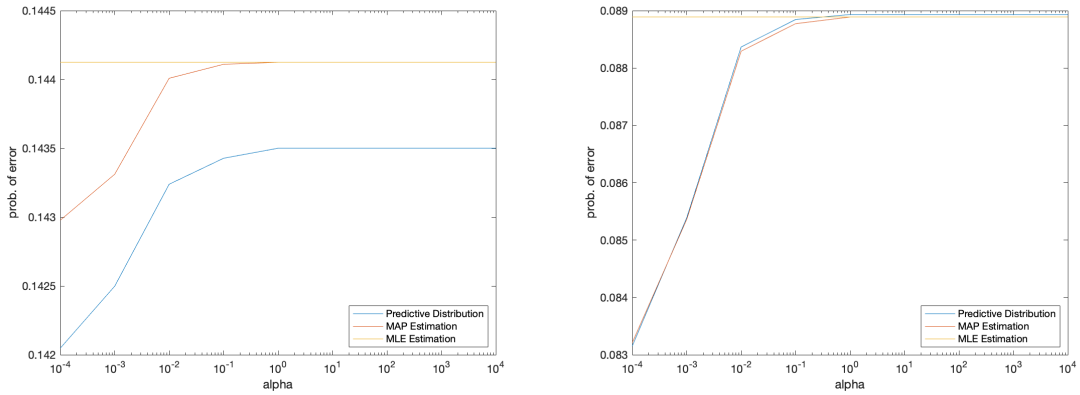

Figure 1: Left: Strategy 1 in Dataset 1, Right: Strategy 1 in Dataset 2

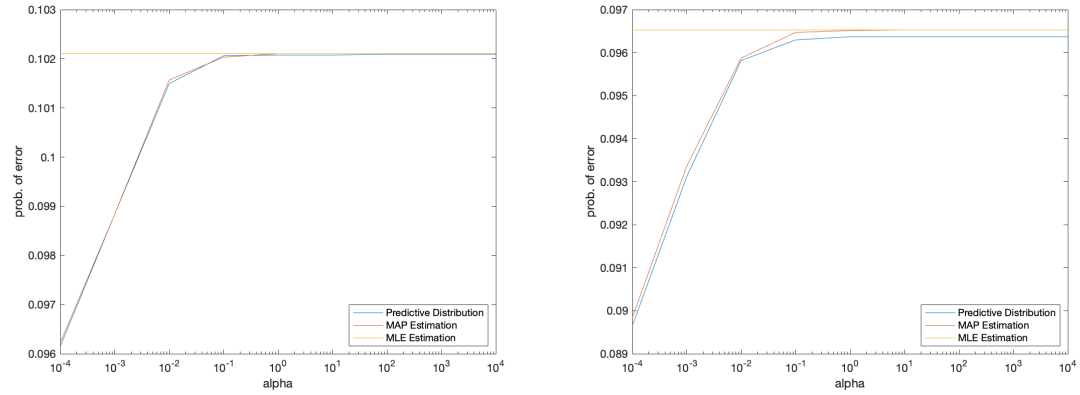

Figure 2: Left: Strategy 1 in Dataset 3, Right: Strategy 1 in Dataset 4

2.1 Relative Behavior of the Three Curves & How they changes from dataset to dataset

(a) **Overall behavior of three curves:** MLE is always a horizontal line since it has nothing to do with the value of α. MAP & predictive distribution error increases with the increasing α and finally converge to MLE (when dataset is big enough). The predictive distribution is slightly better than MAP because it doesn't lose any information of the posterior distribution. PoE result of ML is the worst.

(b) **Comparison of the three methods with different dataset:** When dataset is big, MAP and predictive distribution will be converged to MLE. However, when the dataset is small, the predictive distribution is way much better than MAP and we can't see

predictive distribution converges to MLE. That's because:

$$\mu_1 = \Sigma_0(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}(\frac{1}{n}\sum_{i=1}^{n} x_i) + \frac{1}{n}\Sigma(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}\mu_0$$

If n approaches ∞, we have $\mu_{pred} = \mu_1 = \frac{1}{n}\sum_{i=1}^{n} x_i = \mu_{ML}$

But when n is small $\Sigma_1 = \Sigma_0(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}\Sigma\frac{1}{n}$ will be big

MAP will be really different from predictive distribution

Classification errors of the four datasets are

$$PoE(D_2) < PoE(D_4) < PoE(D_3) < PoE(D_1)$$

although theoretically we should get less error when n increases. This is because that the sizes of the four datasets are not sufficient large so it's reasonable that some of the bigger datasets contain some outliers that making the error increases.

(c) **Comparison of the three methods with same dataset:** MAP and MLE, in the first strategy the probability of error goes higher when α gets larger. That is, the covariance $(\Sigma_0)_{ii} = \alpha w_i$ increases or say when prior becomes less significant. As α gets larger, they will finally converged to MLE if the dataset is big enough. That's because:

$$\mu_1 = \Sigma_0(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}(\frac{1}{n}\sum_{i=1}^{n} x_i) + \frac{1}{n}\Sigma(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}\mu_0$$

$$\Sigma_1 = \Sigma_0(\Sigma_0 + \frac{1}{n}\Sigma)^{-1}\Sigma\frac{1}{n}$$

If Σ_0 approaches 0, we have $\mu_1 = \mu_0$

If $\Sigma_0 >> \Sigma$, we have $\mu_1 = \frac{1}{n}\sum_{i=1}^{n} x_i = \mu_{ML}$

This also means that the prior is good since when α is small we get better result.

3 Strategy 2

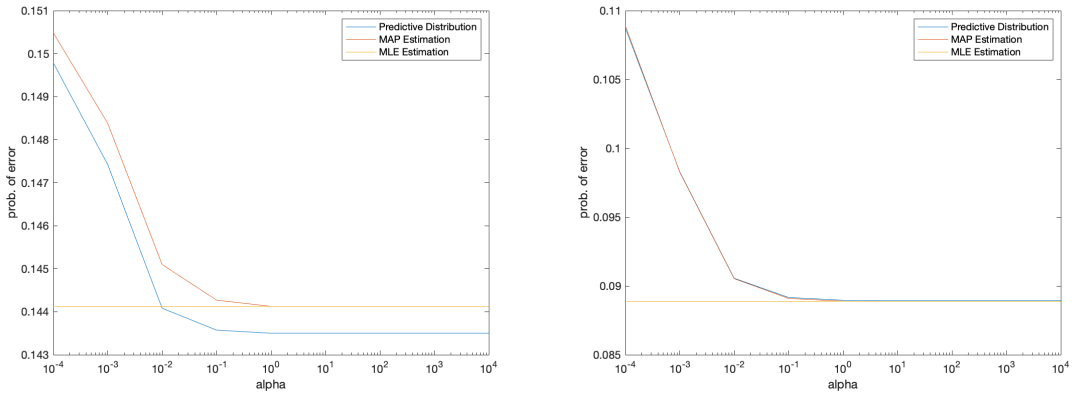

Figure 3: Left: Strategy 2 in Dataset 1, Right: Strategy 2 in Dataset 2

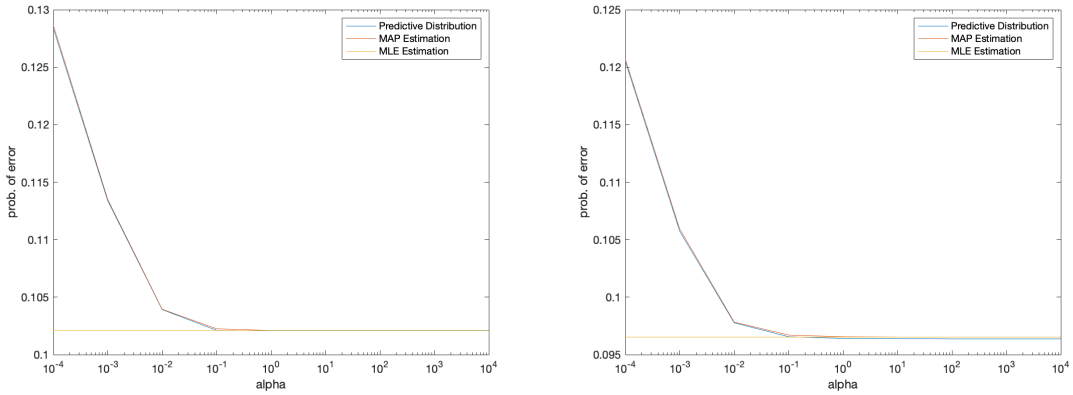

Figure 4: Left: Strategy 2 in Dataset 3, Right: Strategy 2 in Dataset 4

3.1 Relative Behavior of the Three Curves & How they changes from dataset to dataset

(a) **Overall behavior of three curves:** MLE is always a horizontal line since it has nothing to do with the value of α. MAP & predictive distribution error decreases with the increasing α and finally converge to MLE (when dataset is big enough). ML result is the best under strategy 2. PoE result of predictive distribution is the worst.

(b) **Comparison of the three methods with different dataset:** When dataset is big, MAP and predictive distribution will be converged to MLE. However, when the dataset is small, the predictive distribution is way much better than MAP and we can't see predictive distribution converges to MLE. This is the same as strategy 1. In addition, classification errors of the four datasets are

$$PoE(D_2) < PoE(D_4) < PoE(D_3) < PoE(D_1)$$

although theoretically we should get less error when n increases. This is because that the sizes of the four datasets are not sufficient large so it's reasonable that some of the bigger datasets contain some outliers that making the error increases.

(c) **Comparison of the three methods with same dataset:** MAP and MLE, in the first strategy the probability of error goes lower when α gets larger. That is, the covariance $(\Sigma_0)_{ii} = \alpha w_i$ increases or say when prior becomes less significant. As α gets larger, they will finally converged to MLE if the dataset is big enough. The deduction is the same as in strategy 1. This also means that the prior is bad since when α is small we get worse result.

(d) **Others:** It's interesting that when dataset is small, predictive distribution even gets a better result than ML solution under a bad prior. It is reasonable since the dataset is really small and the distribution is not general at all. That's also telling us to use larger dataset to avoid bad fitting.

3.2 Changes of Curves when Strategy Change

When we change from strategy 1 to strategy 2, we see that ML solution changes from the worst to the best. That's because both predictive distribution and MAP solution take prior into consideration. In strategy 1 our prior is reasonable while it's useless in strategy 2. The prior distribution in strategy 2 becomes a burden to the prediction and that's why the ML solution becomes the best.
Codes are attached as follow.

```
% PREDICTIVE_EQUATION.M
load('TrainingSamplesDCT_subsets_8.mat')
load('Alpha.mat')
load('Prior_2.mat') % CHANGE TO PRIOR_2 TO CHECK 2ND STRATEGY

[rg, cg] = size(D1_BG); % CHANGE TO D2, D3 OR D4 FOR OTHER DATASETS
[rc, cc] = size(D1_FG);

% MLE PRIOR
prior_BG = rg / (rc + rg);
prior_FG = rc / (rc + rg);

mu_BG = sum(D1_BG) / rg;
mu_FG = sum(D1_FG) / rc;

sigma_BG = cov(D1_BG);
sigma_FG = cov(D1_FG);

pd_alpha_err = [];
for i = 1:9
    a = alpha(i);
    Sigma0 = diag(a * W0);

    % POSTERIOR MEAN \PROP P(D|MU)P(MU)
    mu1_BG = Sigma0 * inv(Sigma0 + sigma_BG / rg) * mu_BG.' + sigma_BG * inv(Sigma0 +
        sigma_BG / rg) * mu0_BG.' / rg;
    mu1_FG = Sigma0 * inv(Sigma0 + sigma_FG / rc) * mu_FG.' + sigma_FG * inv(Sigma0 +
        sigma_FG / rc) * mu0_FG.' / rc;

    sigma1_BG = Sigma0 * inv(Sigma0 + sigma_BG / rg) * sigma_BG / rg;
    sigma1_FG = Sigma0 * inv(Sigma0 + sigma_FG / rc) * sigma_FG / rc;

    % PREDITIVE DISTRIBUTION PARAMETERS
    pred_mu_BG = mu1_BG.';
    pred_mu_FG = mu1_FG.';

    pred_sigma_BG = sigma1_BG + sigma_BG;
    pred_sigma_FG = sigma1_FG + sigma_FG;

    err = segment_cheetah(prior_FG, prior_BG, pred_mu_FG, pred_mu_BG, pred_sigma_FG,
        pred_sigma_BG, i);
    pd_alpha_err = [pd_alpha_err err];
end

semilogx(alpha, pd_alpha_err)
```

```
% MAP.M
load('TrainingSamplesDCT_subsets_8.mat')
load('Alpha.mat')
load('Prior_2.mat') % CHANGE TO PRIOR_2 TO CHECK 2ND STRATEGY

```

```
[rg, cg] = size(D1_BG); % Change to D2, D3 or D4 for other datasets
[rc, cc] = size(D1_FG);

% MLE Prior
prior_BG = rg / (rc + rg);
prior_FG = rc / (rc + rg);

mu_BG = sum(D1_BG) / rg;
mu_FG = sum(D1_FG) / rc;

sigma_BG = cov(D1_BG);
sigma_FG = cov(D1_FG);

map_alpha_err = [];
for i = 1:9
    a = alpha(i);
    Sigma0 = diag(a * W0);

    % Posterior Mean \prop P(D|mu)P(mu)
    mu1_BG = Sigma0 * inv(Sigma0 + sigma_BG / rg) * mu_BG.' + sigma_BG * inv(Sigma0 +
        sigma_BG / rg) * mu0_BG.' / rg;
    mu1_FG = Sigma0 * inv(Sigma0 + sigma_FG / rc) * mu_FG.' + sigma_FG * inv(Sigma0 +
        sigma_FG / rc) * mu0_FG.' / rc;

    sigma1_BG = Sigma0 * inv(Sigma0 + sigma_BG / rg) * sigma_BG / rg;
    sigma1_FG = Sigma0 * inv(Sigma0 + sigma_FG / rc) * sigma_FG / rc;

    err = segment_cheetah(prior_FG, prior_BG, mu1_FG.', mu1_BG.', sigma_FG, sigma_BG,
        i);
    map_alpha_err = [map_alpha_err err];
end

semilogx(alpha, map_alpha_err)
```

```
% mle.m
load('TrainingSamplesDCT_subsets_8.mat')
load('Alpha.mat')
load('Prior_2.mat') % Change to Prior_2 to check 2nd strategy

[rg, cg] = size(D1_BG); % Change to D2, D3 or D4 for other datasets
[rc, cc] = size(D1_FG);

% MLE Prior
prior_BG = rg / (rc + rg);
prior_FG = rc / (rc + rg);

mu_BG = sum(D1_BG) / rg;
mu_FG = sum(D1_FG) / rc;

sigma_BG = cov(D1_BG);
sigma_FG = cov(D1_FG);

```

```
mle_alpha_err = [];
err = segment_cheetah(prior_FG, prior_BG, mu_FG, mu_BG, sigma_FG, sigma_BG, 1);
for i = 1:9
    mle_alpha_err = [mle_alpha_err err];
end

semilogx(alpha, mle_alpha_err)
```

```
% SEGMENT_CHEETAH.M
function output = segment_cheetah(Pc, Pg, mu_c, mu_g, sigma_c, sigma_g, k)
    cheetah_original = imread('cheetah.bmp');
    [r, c] = size(cheetah_original);
    cheetah_original = im2double(cheetah_original);
    cheetah = padarray(cheetah_original, [4, 4], 'replicate', 'both');

    res = [];
    for i = 5:r+4
        tmp = [];
        for j = 5:c+4
            area = cheetah([i - 3:i + 4], [j - 3:j + 4]);
            dct_res = dct2(area);
            feat = zigzag(dct_res);

            % CALCULATE D(F, C) & D(G, C)
            d_cheetah = (feat - mu_c) * inv(sigma_c) * (feat - mu_c).' + log(det(
                sigma_c)) - 2 * log(Pc);
            d_grass = (feat - mu_g) * inv(sigma_g) * (feat - mu_g).' + log(det(sigma_g
                )) - 2 * log(Pg);

            if d_cheetah >= d_grass
                tmp = [tmp 0];
            else
                tmp = [tmp 255];
            end
        end
        res = [res; tmp];
    end
    figure(k)
    img = imagesc(res);

    mask = imread("cheetah_mask.bmp");
    error = 0; total = r * c;
    for i = 1:r
        for j = 1:c
            mask(i, j);
            if mask(i, j) ~= res(i, j)
                error = error + 1;
            end
        end
    end

    error_rate = error / total
```

```
    output = error_rate;
end
```

```
% ZIGZAG.M
function output = zigzag(in)
    A = load('Zig-Zag Pattern.txt');
    output = [1:64];
    for i = 1:8
        for j = 1:8
            output(A(i, j) + 1) = in(i, j);
        end
    end
end
```