ECE 271A - Statistical Learning 1 - Homework 3

Shubhan Mital

24th November 2025

Notation: $c \in \{\text{FG}, \text{BG}\}$ (class) $\mathbf{x}$ (feature vector) N_c (class sample number) μ_c (class mean) Σ_c (class covariance) $\mu_{0,c}, \Sigma_{0,c}$ (prior mean/covariance) α (prior strength scaling) P_e (probability of error)

(a) Bayesian Setup, Posterior & Predictive Distribution: Strategy 1, D1

Class-Conditional Model: We model $\mathbf{x} \mid Y = c \sim \mathcal{N}(\mu_c, \Sigma_c)$

Empirical ML Estimates:
For class c (FG, BG):

$$\hat{\mu}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \mathbf{x}_i^{(c)} \tag{1}$$

$$\hat{\Sigma}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \left(\mathbf{x}_i^{(c)} - \hat{\mu}_c\right) \left(\mathbf{x}_i^{(c)} - \hat{\mu}_c\right)^T \tag{2}$$

Likelihood for All Samples:

$$p(\mathcal{D}_c|\mu_c) = \prod_{i=1}^{N_c} \mathcal{N}(\mathbf{x}_i^{(c)}; \mu_c, \hat{\Sigma}_c)$$

Prior on Mean (μ_c):

$$\mu_c \sim \mathcal{N}(\mu_{0,c},\, \alpha^{-1} W), \tag{3}$$

$$\mu_{0,FG} = 1\,(\text{DC}),\ 0 \text{ elsewhere}, \tag{4}$$

$$\mu_{0,BG} = 3\,(\text{DC}),\ 0 \text{ elsewhere}, \tag{5}$$

$$W = \text{diag}(w_1, \ldots, w_{64}) \tag{6}$$

Posterior of Mean Derivation:
Given Gaussian prior on mean and Gaussian likelihood:

$$p(\mu_c|\mathcal{D}_c) \propto p(\mathcal{D}_c|\mu_c)\, p(\mu_c)$$

$$\log p(\mu_c|\mathcal{D}_c) = -\frac{1}{2} \sum_{i=1}^{N_c} (\mathbf{x}_i^{(c)} - \mu_c)^T \hat{\Sigma}_c^{-1} (\mathbf{x}_i^{(c)} - \mu_c) - \frac{1}{2} (\mu_c - \mu_{0,c})^T (\alpha^{-1} W)^{-1} (\mu_c - \mu_{0,c}) + C$$

Completing the square in μ_c:

$$\underset{\text{Precision}}{\Sigma_{1,c}^{-1}} = \underbrace{\alpha W^{-1}}_{\text{prior precision}} + \underbrace{N_c \hat{\Sigma}_c^{-1}}_{\text{data precision}}$$

$$\underset{\text{Posterior Mean}}{\mu_{1,c}} = \Sigma_{1,c} \left[\alpha W^{-1} \mu_{0,c} + \hat{\Sigma}_c^{-1} \sum_{i=1}^{N_c} \mathbf{x}_i^{(c)} \right]$$

Let $\sum_{i=1}^{N_c} \mathbf{x}_i^{(c)} = N_c\hat{\mu}_c$:

$$\boxed{\mu_{1,c} = \Sigma_{1,c}\left[\alpha W^{-1}\mu_{0,c} + N_c\hat{\Sigma}_c^{-1}\hat{\mu}_c\right]}$$

Interpretation: - α small $\rightarrow$ data ignored, posterior shrinks to prior; - α large $\rightarrow$ prior ignored, posterior equals ML mean.

Predictive Distribution:
Bayesian predictive marginalizes μ_c:

$$p(\mathbf{x}|\mathcal{D}_c) = \int p(\mathbf{x}|\mu_c, \hat{\Sigma}_c)p(\mu_c|\mathcal{D}_c)d\mu_c$$

As both are Gaussian, the result is:

$$p(\mathbf{x}|\mathcal{D}_c) = \mathcal{N}(\mu_{1,c}, \hat{\Sigma}_c + \Sigma_{1,c})$$

Justification: Predictive variance equals data variance plus parameter uncertainty (covariance in mean estimate).

ML Priors:

$$P(Y = c) = \frac{N_c}{N_{FG} + N_{BG}}$$

Bayesian Decision Rule (log discriminant): For each 8×8 block with DCT feature $\mathbf{x}$:

$$g_c(\mathbf{x}) = \log P(Y = c) - \frac{1}{2}\log|\hat{\Sigma}_c + \Sigma_{1,c}| - \frac{1}{2}(\mathbf{x} - \mu_{1,c})^T(\hat{\Sigma}_c + \Sigma_{1,c})^{-1}(\mathbf{x} - \mu_{1,c})$$

Assign to class with largest $g_c(\mathbf{x})$.

Probability of Error:

$$P_e = \frac{\#\text{misclassified blocks}}{\text{total blocks}}$$

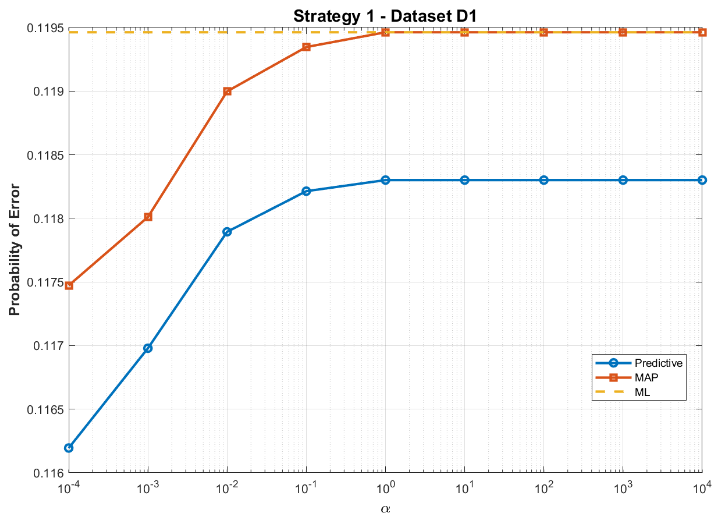

Figure 1: Strategy 1, D1: Probability of Error vs. α for ML, MAP, Predictive

Analysis: Small α: strong prior, lowest error. Mid-range α: prior weakening, error rises. Large α: prior lost, predictive reduces to ML.

(b) Maximum Likelihood Estimation (ML): Strategy 1, D1

Step 1: ML Parameter Estimation

Given N_c training samples $\{\mathbf{x}_i^{(c)}\}_{i=1}^{N_c}$ for class $c \in \{\mathrm{FG}, \mathrm{BG}\}$, the ML estimators for the class mean and covariance are:

$$\hat{\mu}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \mathbf{x}_i^{(c)} \tag{7}$$

$$\hat{\Sigma}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \left(\mathbf{x}_i^{(c)} - \hat{\mu}_c\right) \left(\mathbf{x}_i^{(c)} - \hat{\mu}_c\right)^T \tag{8}$$

These arise by maximizing the log-likelihood:

$$\log L(\mu_c, \Sigma_c) = \sum_{i=1}^{N_c} \log \mathcal{N}(\mathbf{x}_i^{(c)}; \mu_c, \Sigma_c) = -\frac{N_c}{2} \log |\Sigma_c| - \frac{1}{2} \sum_{i=1}^{N_c} (\mathbf{x}_i^{(c)} - \mu_c)^T \Sigma_c^{-1} (\mathbf{x}_i^{(c)} - \mu_c) + \text{const} \tag{9}$$

Setting derivatives to zero yields the above ML estimates.

Step 2: ML Class-Conditional Density

The class-conditional probability for a test vector $\mathbf{x}$ under class c:

$$p(\mathbf{x}|c) = \frac{1}{(2\pi)^{d/2} |\hat{\Sigma}_c|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \hat{\mu}_c)^T \hat{\Sigma}_c^{-1} (\mathbf{x} - \hat{\mu}_c)\right)$$

where $d = 64$ (DCT features).

Step 3: Class Priors

Empirical class priors are estimated as:

$$P(Y = c) = \frac{N_c}{N_{FG} + N_{BG}}$$

Step 4: Discriminant Function and Classification

The log-posterior discriminant for each class is:

$$g_{ML,c}(\mathbf{x}) = \log P(Y = c) - \frac{1}{2} \log |\hat{\Sigma}_c| - \frac{1}{2} (\mathbf{x} - \hat{\mu}_c)^T \hat{\Sigma}_c^{-1} (\mathbf{x} - \hat{\mu}_c)$$

Classify $\mathbf{x}$ as $\arg\max_c \, g_{ML,c}(\mathbf{x})$.

Step 5: Error Calculation

Apply the classifier to all evaluation blocks and determine the error rate:

$$P_e = \frac{\#\text{ misclassified blocks}}{\text{total blocks}}$$

Step 6: Behavior and Limitations

- ML makes no use of prior knowledge; it is only reliable for large sample sizes N_c.
- For small N_c, $\hat{\mu}_c$ and $\hat{\Sigma}_c$ can be noisy, causing overfitting and high variance in P_e.
- ML is equivalent to Bayesian classification with a non-informative (flat) prior.

Relation to HW2: This is the same as the classical plug-in Gaussian classifier developed in assignment 2, thus serving as a baseline for Bayesian experiments.

(c) Maximum a Posteriori (MAP) Estimation: Strategy 1, D1

Step 1: MAP Parameter Derivation
Recall the posterior distribution for the mean (from part (a)):

$$p(\mu_c|\mathcal{D}_c) = \mathcal{N}(\mu_{1,c}, \Sigma_{1,c}) \qquad \Sigma_{1,c}^{-1} = \alpha W^{-1} + N_c\hat{\Sigma}_c^{-1} \qquad \mu_{1,c} = \Sigma_{1,c}\left[\alpha W^{-1}\mu_{0,c} + N_c\hat{\Sigma}_c^{-1}\hat{\mu}_c\right] \tag{10}$$

The MAP estimate is the mode of this posterior, i.e., $\mu_{MAP,c} = \mu_{1,c}$.

Step 2: Interpretation of MAP Formula
- For small N_c, MAP pulls $\hat{\mu}_c$ strongly toward the prior mean $\mu_{0,c}$
- For large N_c, the influence of the prior vanishes: $\mu_{MAP,c} \to \hat{\mu}_c$.
- The relative weights are determined by α (prior strength) and N_c (data amount).

Step 3: MAP Class-Conditional Density
The MAP classifier uses point estimate $\mu_{MAP,c}$ in the Gaussian:

$$p(\mathbf{x}|c) = \mathcal{N}(\mathbf{x};\, \mu_{MAP,c},\, \hat{\Sigma}_c)$$

MAP does not account for the uncertainty (variance) in the mean, as opposed to the predictive Bayesian.

Step 4: MAP Discriminant Function

$$g_{MAP,c}(\mathbf{x}) = \log P(Y=c) - \frac{1}{2}\log|\hat{\Sigma}_c| - \frac{1}{2}(\mathbf{x} - \mu_{MAP,c})^T\hat{\Sigma}_c^{-1}(\mathbf{x} - \mu_{MAP,c})$$

Again, assign the class by maximizing $g_{MAP,c}(\mathbf{x})$.

Step 5: Extreme Cases and Comparison to ML

- When $\alpha \to 0$ (very strong prior): $\mu_{MAP,c} \to \mu_{0,c}$ (fully prior-driven).
- When $\alpha \to \infty$ (very weak prior): $\mu_{MAP,c} \to \hat{\mu}_c$ (reduces to ML).
- For intermediate α, $\mu_{MAP,c}$ is a data/prior weighted average.

Step 6: Source of MAP Regularization
MAP regularizes the sample mean more strongly when the number of training samples N_c is small, acting as shrinkage toward prior knowledge.

Step 7: Error Computation

$$P_e = \frac{\text{\# misclassified blocks}}{\text{total blocks}}$$

as in ML and Bayesian predictive cases.

Step 8: Limits of MAP (vs. Predictive) - MAP does not integrate over posterior uncertainty. Predictive Bayesian can outperform MAP, especially when prior is helpful and data limited.

Summary: MAP interpolates between ML and prior; only predictive classifier adds posterior uncertainty to covariance.

(d) Results Across All Dataset Sizes (Strategy 1)

Dataset Sizes:

	D1	D2	D3	D4
FG (cheetah)	75	125	175	225
BG (grass)	300	500	700	900

Strategy 1 Probability of Error

- D1: ML $P_e = 0.1195$, Predictive $P_e = 0.1162$, MAP $P_e = 0.1175$
- D2: ML $P_e = 0.0725$, Predictive $P_e = 0.0671$, MAP $P_e = 0.0671$
- D3: ML $P_e = 0.0838$, Predictive $P_e = 0.0781$, MAP $P_e = 0.0782$
- D4: ML $P_e = 0.0782$, Predictive $P_e = 0.0722$, MAP $P_e = 0.0724$

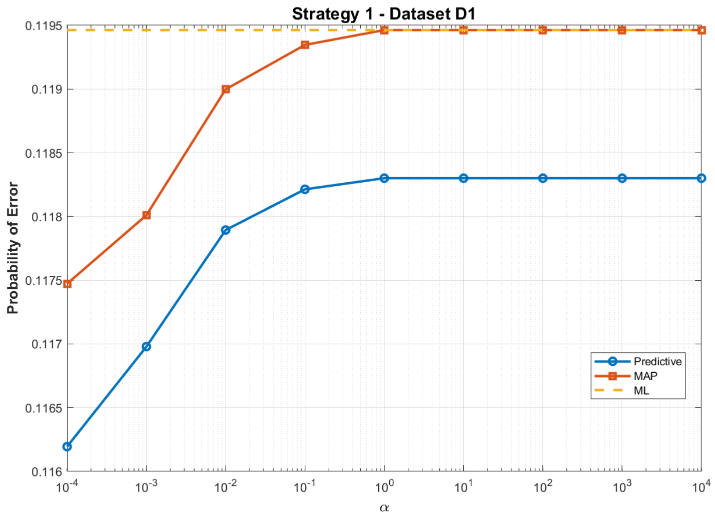

Figure 2: Strategy 1 - Dataset D1

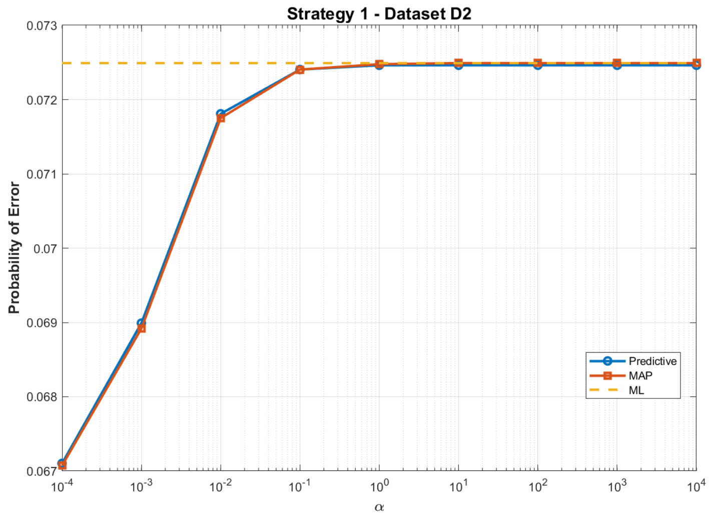

Figure 3: Strategy 1 - Dataset D2

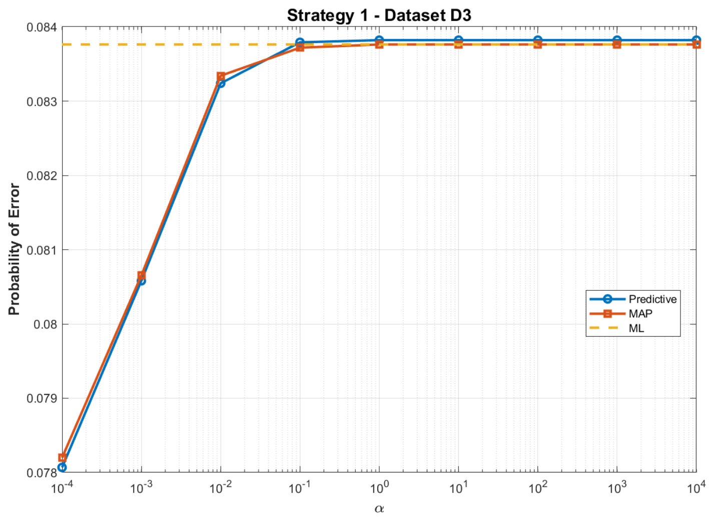

Figure 4: Strategy 1 - Dataset D3

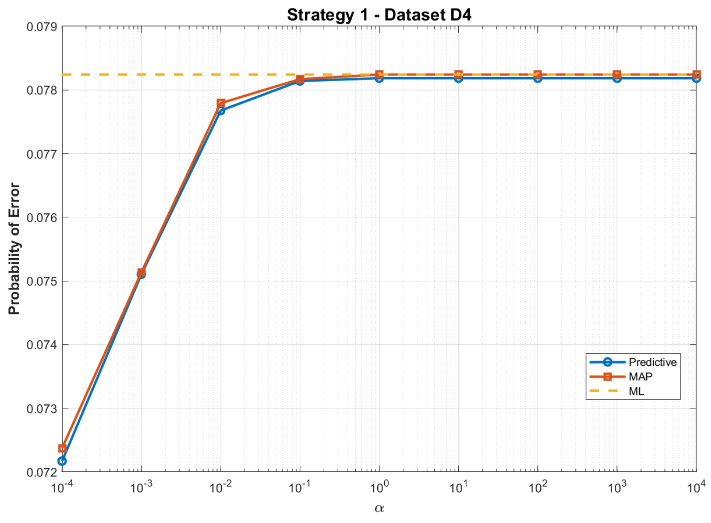

Figure 5: Strategy 1 - Dataset D4

(e) Strategy 2: Class-Neutral Priors, Mathematics, and Analysis

Prior Setup: Strategy 2 uses a neutral prior mean for DC coefficient (mid-range of observed amplitudes for both classes).

$$\mu_{0,FG} = \mu_{0,BG} = 2$$

Strategy 2 Probability of Error

- D1: ML $P_e = 0.1195$, Predictive $P_e = 0.1183$, MAP $P_e = 0.1195$
- D2: ML $P_e = 0.0725$, Predictive $P_e = 0.0725$, MAP $P_e = 0.0725$
- D3: ML $P_e = 0.0838$, Predictive $P_e = 0.0838$, MAP $P_e = 0.0838$
- D4: ML $P_e = 0.0782$, Predictive $P_e = 0.0782$, MAP $P_e = 0.0782$

Figures:

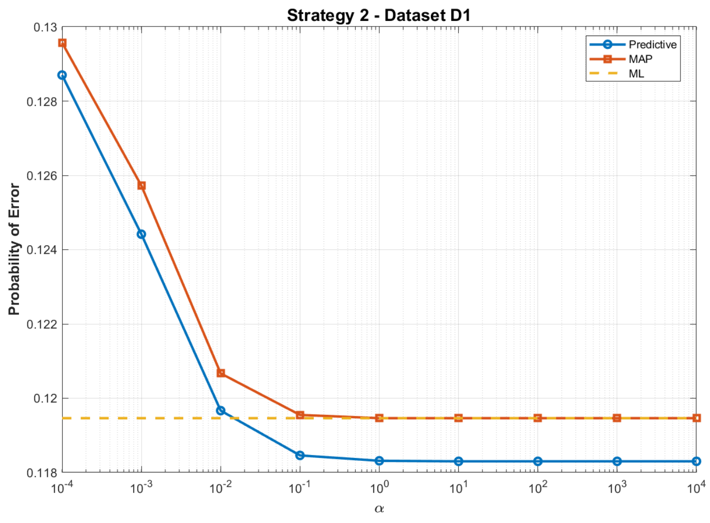

Figure 6: Strategy 2 - Dataset D1

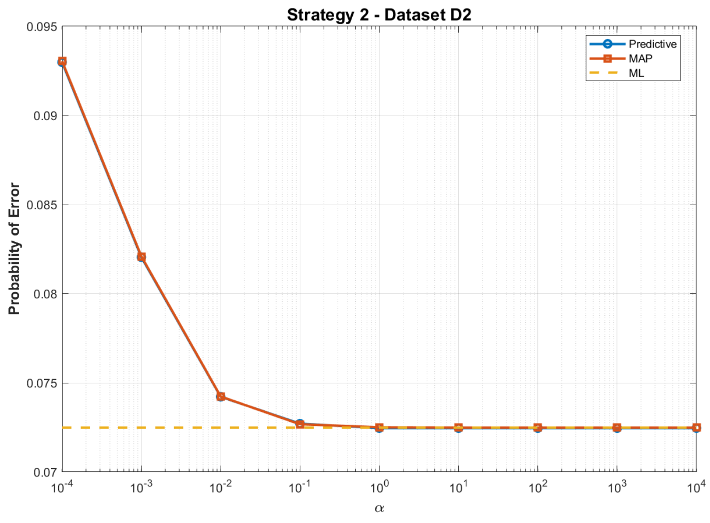

Figure 7: Strategy 2 - Dataset D2

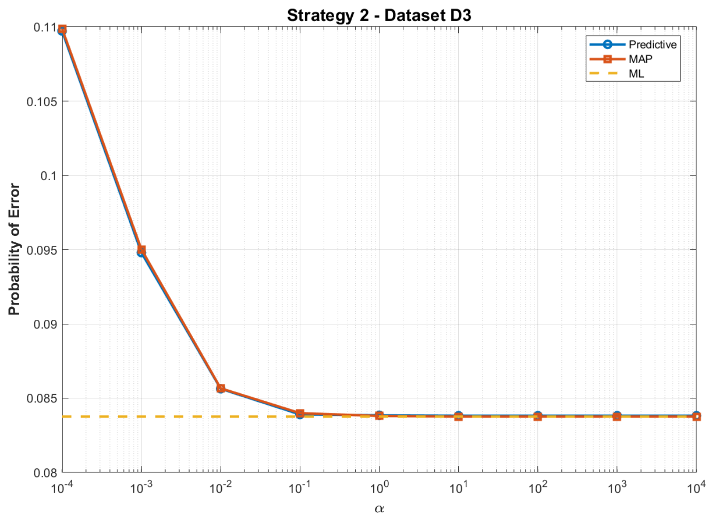

Figure 8: Strategy 2 - Dataset D3

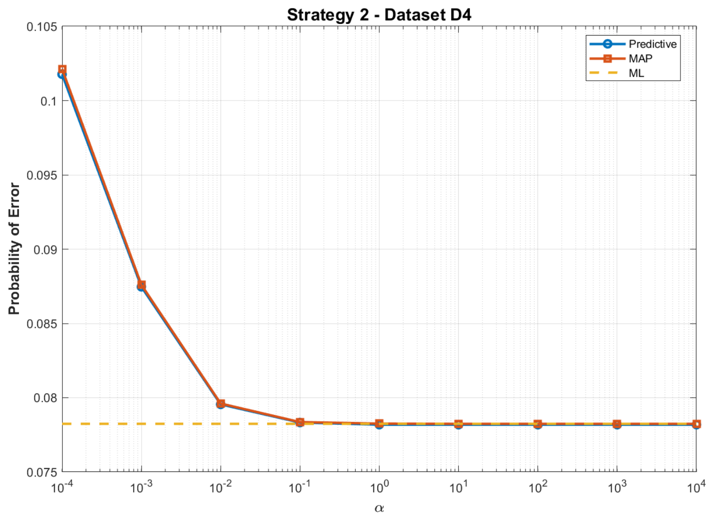

Figure 9: Strategy 2 - Dataset D4

Comparative Analysis and Theoretical Insights

In this section we synthesize all the empirical and theoretical findings across both prior strategies, all four datasets, and the three classifiers (ML, MAP, predictive Bayesian). We aim to rigorously explain both qualitative and quantitative trends measured in probability of error.

1. Strategy 1 vs. Strategy 2

Strategy 1: A class-dependent prior: for the foreground (cheetah), the prior mean on the DC coefficient is small (representing darker blocks); for the background (grass), it is large (lighter blocks). This prior leverages real statistical differences between cheetah and grass and this strategy allows Bayesian methods (predictive, MAP) to push mean estimates toward the respective class traits, which is particularly beneficial when data is scarce.

Strategy 2: A neutral prior: both class priors for the DC term are set at the midpoint between cheetah and grass block mean values. Thus, prior information no longer offers discrimination between classes. Therefore this prior only functions as a regularizer. It cannot steer the mean estimates toward a "better" value for classification, and if set too strong, it can actually degrade separation.

2. Dataset Size Effects: D1 vs D2 vs D3 vs D4

The four datasets represent progressively larger training sample sizes for both foreground (cheetah) and background (grass). We observe that, for all classifier methods and prior strategies, the probability of error P_e follows a non-monotonic trend: D1 exhibits the highest error, D2 achieves the lowest error, while D3 and D4 show higher error than D2. To understand this, consider the following:

- **D1 (Smallest Training Set):** In D1, both mean and covariance estimates are highly unstable due to limited training data. This high estimation variance leads to large uncertainty in the resulting class-conditional densities-most pronounced for the ML classifier, since it is entirely data-driven and lacks any regularization. For Bayesian methods, MAP and predictive Bayesian provide some mitigation by drawing on prior structure, but the meager data means the prior cannot fully overcome the variance, especially if it is not perfectly informative.
  - Under *Strategy 1* (informative prior), the prior helps by biasing estimates in a useful direction, resulting in a lower P_e relative to ML.
  - Under *Strategy 2* (neutral prior), the prior does not differentiate between classes, and can actively harm performance for small α by forcing class means to overlap.
- **D2 (Intermediate):** This represents a sweet spot for the Bayesian paradigm. The training set is now large enough to allow reliable mean and covariance estimation, but still small enough that the prior can meaningfully regularize parameters and reduce variance. As a result, classifier performance is maximally improved-probability of error is lowest, especially for predictive Bayesian (if the prior is informative). This is the regime where Bayesian learning most powerfully balances data and prior information.
- **D3 and D4 (Largest Sets):** As the number of samples continues to grow, the influence of the prior diminishes-the data increasingly dominates mean and covariance estimation, regardless of prior information. While increased data generally improves parameter estimation (reducing bias), changes in block composition may mean that more ambiguous or overlapping blocks enter the training set. This can slightly increase in-class variance and cause P_e to rise relative to D2, as decision boundaries become less distinct. Both Bayesian and ML methods now converge, and any prior structure becomes nearly irrelevant.

Key Points Across Strategies and Methods:

- For the *informative prior* (Strategy 1), Bayesian methods outperform ML strongly for D1/D2, with predictive $<$ MAP $<$ ML. For D3/D4, all methods converge in performance.
- For the *neutral prior* (Strategy 2), there is no benefit from Bayesian regularization; in fact, for D1, Bayesian P_e can be higher than ML for small α (strong prior), since the prior collapses class separation.
- The overall non-monotonic shape (minimum at D2) is due to the interplay of estimation variance, prior regularization, and the possible inclusion of more ambiguous data as the sample size increases. It demonstrates that "more data" does not always straightforwardly mean "less error"-the nature and informativeness of both data and prior matter just as much. Even at the largest datasets, it is crucial to recognize that parameter uncertainty becomes negligible: thus both MAP and predictive Bayesian classifiers collapse to the ML solution.

3. ML vs MAP vs Predictive: Performance and Theory

- **ML**: Maximizes the likelihood without regard for uncertainty or prior information.
  - Pros: Optimal if the dataset is very large.
  - Cons: Extremely sensitive to small training set size; prone to overfitting.
- **MAP**: Regularizes sample means by shrinking them toward the prior mean.
  - Pros: Reduces variance when N_c is small.
  - Cons: Introduces bias if the prior is incorrect; limited improvement for large datasets.
- **Predictive Bayesian**: Integrates over uncertainty in the mean to produce an effective covariance.
  - Pros: Best performance in very small-data settings; accounts for parameter uncertainty.
  - Cons: Converges to MAP/ML as data increases.

Why do these trends arise?
Small α (strong prior): When the prior is highly informative (Strategy 1), Bayesian methods can substantially improve estimation of class means (reducing bias and variance). The predictive method performs best, as it incorporates mean uncertainty into decision boundaries. If the prior is neutral or misleading (Strategy 2), this regularization may harm performance, forcing classes to overlap more.

Medium α (balanced prior): Bayesian methods optimally combine prior information and data, minimizing error.

Large α (weak prior): Prior becomes negligible; all methods reduce to ML.

Appendix: Matlab Code

```
%% =================Dataset=====================================================
%  ECE 271A - HW3: Bayesian Classification with Posterior Predictive & MAP
%  Author: Shubhan Mital
%  Description: Implements cheetah vs. grass segmentation on a grayscale image using↙
Bayesian classification rules based on DCT features extracted from 8x8 image blocks.
%  Trains class-conditional multivariate Gaussian models using provided datasets, then↙
performs pixel-wise classification using three strategies:
%        - ML (Maximum Likelihood)
%        - MAP (Maximum a Posteriori, learned mean with empirical covariance)
%        - Predictive posterior (Bayesian integration, diagonal prior covariance and↙
alpha regularization sweep)
%  For each dataset and prior, computes the Probability of Error (PE) against the↙
ground-truth mask, and visualizes PE versus alpha for each approach.
%  ===================================================================
clear; close all; clc;
warning('off', 'all');
output_path = "C:\Users\shubh\Desktop\Hard disk\College(PG)\Academics at UCSD\Y1Q1\ECE↙
271A - Statistical Learning 1\Homework\homework3\Output images";

% ---------------------------Load Data & Priors---------------------------
data = load('TrainingSamplesDCT_subsets_8.mat');
alpha = load('Alpha.mat');
prior1 = load('Prior_1.mat');
prior2 = load('Prior_2.mat');
load('Alpha.mat');
load('TrainingSamplesDCT_subsets_8.mat');
cheetah_img = get_processed_image('cheetah.bmp');
cheetah_mask = get_processed_image('cheetah_mask.bmp');

% extract priors for two strategies
mu0_FG_strat1 = prior1.mu0_FG;
mu0_BG_strat1 = prior1.mu0_BG;
w_strat1 = prior1.W0;

mu0_FG_strat2 = prior2.mu0_FG;
mu0_BG_strat2 = prior2.mu0_BG;
w_strat2 = prior2.W0;

% ---------------------------Setup datasets---------------------------
datasets = {D1_FG, D1_BG; D2_FG, D2_BG; D3_FG, D3_BG; D4_FG, D4_BG};
dataset_names = {'D1', 'D2', 'D3', 'D4'};
num_alphas = length(alpha);

% Precompute image features (DCT + zigzag) for all 8x8 blocks
[img_height, img_width] = size(cheetah_img);
num_blocks_row = img_height - 7;
num_blocks_col = img_width - 7;
nblocks = num_blocks_row * num_blocks_col;
image_features = extract_image_features(cheetah_img);
```

```
% ground truth full mask (0 or 255)
ground_truth_full = double(cheetah_mask);
num_pixels_full = numel(ground_truth_full);

if max(ground_truth_full(:)) == 1
    ground_truth_full = ground_truth_full * 255;
end

% numeric regularization constant
eps_reg = 1e-6;

% Loop over strategies and datasets
for strategy = 1:2
    fprintf('\n========== STRATEGY %d ==========\n', strategy);
    if strategy == 1
        mu0_FG = prior1.mu0_FG(:);
        mu0_BG = prior1.mu0_BG(:);
        W_vec = prior1.W0(:);
    else
        mu0_FG = prior2.mu0_FG(:);
        mu0_BG = prior2.mu0_BG(:);
        W_vec = prior2.W0(:);
    end

    if numel(W_vec) ~= 64
        W_vec = ones(64,1);
    end

    for d = 1:4
        FG_data = datasets{d, 1};
        BG_data = datasets{d, 2};
        N_FG = size(FG_data, 1);
        N_BG = size(BG_data, 1);
        P_FG = N_FG / (N_FG + N_BG);
        P_BG = N_BG / (N_FG + N_BG);
        mean_FG = mean(FG_data, 1)';
        mean_BG = mean(BG_data, 1)';
        Sigma_FG = cov(FG_data, 1) + eps_reg*eye(64);
        Sigma_BG = cov(BG_data, 1) + eps_reg*eye(64);

        % ML
        pred_mask_ml = classify_image_fast(image_features, mean_BG, Sigma_BG, mean_FG,↙
Sigma_FG, P_BG, P_FG);
        pred_map_ML = map_block_predictions_center(pred_mask_ml, img_height,↙
img_width);
        if max(pred_map_ML(:)) == 1
            pred_map_ML = pred_map_ML * 255;
        end
        PE_ML = sum(pred_map_ML(:) ~= ground_truth_full(:)) / num_pixels_full;
        PE_predictive = zeros(num_alphas,1); PE_MAP = zeros(num_alphas,1);
        for a_idx = 1:num_alphas
            alpha_val = alpha(a_idx);
            Sigma0 = alpha_val * diag(W_vec) + eps_reg * eye(64);

            % --------- Posterior parameters --------
            [mu1_FG, Sigma1_FG] = compute_posterior_parameters(mean_FG, Sigma_FG,↙
mu0_FG, Sigma0, N_FG);
            [mu1_BG, Sigma1_BG] = compute_posterior_parameters(mean_BG, Sigma_BG,↙
mu0_BG, Sigma0, N_BG);

            % Predictive
            mu_pred_FG = mu1_FG; Sigma_pred_FG = Sigma_FG + Sigma1_FG;
            mu_pred_BG = mu1_BG; Sigma_pred_BG = Sigma_BG + Sigma1_BG;
            pred_mask_pred = classify_image_fast(image_features, mu_pred_BG,↙
Sigma_pred_BG, mu_pred_FG, Sigma_pred_FG, P_BG, P_FG);
            pred_map_pred = map_block_predictions_center(pred_mask_pred, img_height,↙
img_width);
            if max(pred_map_pred(:)) == 1
                pred_map_pred = pred_map_pred * 255;
            end
            PE_predictive(a_idx) = sum(pred_map_pred(:) ~= ground_truth_full(:)) /↙
num_pixels_full;

            % MAP
            mu_MAP_FG = mu1_FG; mu_MAP_BG = mu1_BG;
            pred_mask_map = classify_image_fast(image_features, mu_MAP_BG, Sigma_BG,↙
mu_MAP_FG, Sigma_FG, P_BG, P_FG);
            pred_map_MAP = map_block_predictions_center(pred_mask_map, img_height,↙
img_width);
            if max(pred_map_MAP(:)) == 1
                pred_map_MAP = pred_map_MAP * 255;
            end
            PE_MAP(a_idx) = sum(pred_map_MAP(:) ~= ground_truth_full(:)) /↙
num_pixels_full;
        end

        % Plot + save
        figure('Position', [100, 100, 900, 600]);
        semilogx(alpha, PE_predictive, '-o', 'LineWidth', 2, 'DisplayName',↙
'Predictive'); hold on;
        semilogx(alpha, PE_MAP, '-s', 'LineWidth', 2, 'DisplayName', 'MAP');
        semilogx(alpha, PE_ML * ones(size(alpha)), '--', 'LineWidth', 2, 'DisplayName',↙
'ML');
        xlabel('\alpha', 'FontSize', 12, 'FontWeight', 'bold');
        ylabel('Probability of Error', 'FontSize', 12, 'FontWeight', 'bold');
        title(sprintf('Strategy %d - Dataset %s', strategy, dataset_names{d}),↙
'FontSize', 14, 'FontWeight', 'bold');
        legend('Location', 'best'); grid on; hold off;
        saveas(gcf, fullfile(output_path, sprintf('Strategy%d_Dataset%s.png', strategy,↙
dataset_names{d})));

        % Print summary
        fprintf('Dataset %s: ML PE=%.4f, Predictive PE=%.4f, MAP PE=%.4f\n\n', ...
            dataset_names{d}, PE_ML, min(PE_predictive), min(PE_MAP));
    end
end
fprintf('\n========== ANALYSIS COMPLETE ==========\n');

% ---------------------------Helper Functions---------------------------
function arr = extract_image_features(img)
    [nrow, ncol] = size(img);
    arr = zeros((nrow-7)*(ncol-7), 64);
    idx = 1;
    for i = 1:(nrow-7)
        for j = 1:(ncol-7)
            block = img(i:i+7, j:j+7);
            arr(idx, :) = zigzag_scan(dct2(block));
            idx = idx + 1;
        end
    end
end

function [mu1, Sigma1] = compute_posterior_parameters(mu_sample, Sigma_sample, mu0,↙
Sigma0, N)
    Sigma0_inv = inv(Sigma0);
    Sigma_inv = inv(Sigma_sample);
    Sigma1 = inv(Sigma0_inv + N * Sigma_inv);
    mu1 = Sigma1 * (Sigma0_inv * mu0 + N * Sigma_inv * mu_sample);
end

function zz = zigzag_scan(block)
    zz_order = [
        1,2,6,7,15,16,28,29;
        3,5,8,14,17,27,30,43;
        4,9,13,18,26,31,42,44;...
       10,12,19,25,32,41,45,54;
       11,20,24,33,40,46,53,55;
       21,23,34,39,47,52,56,61;...
       22,35,38,48,51,57,60,62;
       36,37,49,50,58,59,63,64
       ];
    zz = zeros(1,64);
    for k=1:64
        [i,j]=find(zz_order==k);
        zz(k)=block(i,j);
    end
end

function pred = classify_image_fast(features, mu_BG, Sigma_BG, mu_FG, Sigma_FG,↙
prob_BG, prob_FG)
    try
        L_BG = chol(Sigma_BG, 'lower');
    catch
        Sigma_BG = Sigma_BG + 1e-6*eye(64);
        L_BG = chol(Sigma_BG,'lower');
    end

    try
        L_FG = chol(Sigma_FG, 'lower');
    catch
        Sigma_FG = Sigma_FG + 1e-6*eye(64);
        L_FG = chol(Sigma_FG,'lower');
    end

    logdet_BG = 2*sum(log(diag(L_BG)));
    logdet_FG = 2*sum(log(diag(L_FG)));
    diff_BG = features - mu_BG';
    diff_FG = features - mu_FG';
    z_BG = (L_BG \ diff_BG')';
    z_FG = (L_FG \ diff_FG')';
    ll_BG = -0.5*(64*log(2*pi) + logdet_BG + sum(z_BG.^2,2));
    ll_FG = -0.5*(64*log(2*pi) + logdet_FG + sum(z_FG.^2,2));
    logpost_BG = ll_BG + log(prob_BG);
    logpost_FG = ll_FG + log(prob_FG);
    pred = double(logpost_FG > logpost_BG) * 255;
end

function full_map = map_block_predictions_center(pred, img_height, img_width)
    num_blocks_row = img_height-7;
    num_blocks_col = img_width-7;
    full_map = zeros(img_height,img_width);
    idx = 1;
    for i = 1:num_blocks_row
        for j = 1:num_blocks_col
            full_map(i+3,j+3) = pred(idx);
            idx=idx+1;
        end
    end
end

function img = get_processed_image(path)
    img = imread(path); img = im2double(img);
    if ndims(img)==3, img = rgb2gray(img); end
end
```