Question 1 (50%)

The probability density function (pdf) for a 2-dimensional real-valued random vector $\mathbf{X}$ is as follows: $p(\mathbf{x}) = p(L=0)p(\mathbf{x}|L=0) + p(L=1)p(\mathbf{x}|L=1)$. Here L is the true class label that indicates which class-conditioned pdf generates the data.

Assume the class priors are $p(L=0) = 0.65$ and $p(L=1) = 0.35$. The class-conditional pdfs are also $p(\mathbf{x}|L=0) = a_1\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{01}, \boldsymbol{\Sigma}_{01}) + a_2\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{02}, \boldsymbol{\Sigma}_{02})$ and $p(\mathbf{x}|L=1) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$, where $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a multivariate Gaussian probability density function with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. The parameters of the class-conditional Gaussian pdfs are $a_1 = a_2 = 1/2$, and:

$$\boldsymbol{\mu}_{01} = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \quad \boldsymbol{\Sigma}_{01} = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \quad \boldsymbol{\mu}_{02} = \begin{bmatrix} 0 \\ 3 \end{bmatrix} \quad \boldsymbol{\Sigma}_{02} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \quad \boldsymbol{\mu}_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix} \quad \boldsymbol{\Sigma}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

For numerical results requested below, generate the following independent datasets each consisting of iid samples from the specified data distribution, and in each dataset make sure to include the true class label for each sample.

- D_{train}^{20} consists of 20 samples and their labels for training;
- D_{train}^{200} consists of 200 samples and their labels for training;
- D_{train}^{2000} consists of 2000 samples and their labels for training;
- D_{valid}^{10K} consists of 10000 samples and their labels for validation.

Part 1 (10%): Determine the theoretically optimal classifier that achieves minimum probability of error using knowledge of the true pdf. Specify the classifier mathematically and implement it; then apply it to all samples in D_{valid}^{10K}. From this decision results and true labels for this validation set, estimate and plot the ROC curve of this min-Pr(error) classifier. Report the optimal threshold and probability error estimate of the theoretical min-Pr(error) classifier, indicating on the ROC curve with a special marker its location. Also report the empirical threshold and associated minimum probability of error estimate for this classifier based on counts of decision-truth label pairs on D_{valid}^{10K}.

Optional (Bonus 2.5%): As supplementary visualization, generate a plot of the decision boundary of this classification rule overlaid on the validation dataset. This establishes an aspirational performance level on the dataset for the following approximations.

Part 2 (40%): (a) Using the maximum likelihood parameter estimation technique train three separate logistic-linear approximations of class label posterior functions given a sample. For each approximation use one of the three training datasets D_{train}^{20}, D_{train}^{200}, D_{train}^{2000}. When optimizing the parameters, specify the optimization problem as minimization of the negative-log-likelihood (NLL) of the training dataset, and use your favorite numerical optimization approach, such as gradient descent or Python's optimize.minimize function in the scipy library. Determine how to use these class-label-posterior approximations to classify a sample in order to approximate the minimum-Pr(error) classification rule; apply these three approximations of the class label posterior function on samples in D_{valid}^{10K}, and estimate the probability of error that these three classification rules will attain (using counts of decisions on the validation set).

Optional (Bonus 2.5%): As supplementary visualization, generate plots of the decision boundaries of these trained classifiers superimposed on their respective training datasets and the validation dataset.

(b) Repeat the process described in Part (2a) using a logistic-quadratic-function approximation of class label posterior functions given a sample. How does the performance of your classifiers trained in this part compare to each other considering differences in number of training samples and function form? How do they compare to the theoretically optimal classifier from Part 1? Briefly discuss results and insights.

Note: With $\mathbf{x}$ representing the input sample vector and $\mathbf{w}$ denoting the model weights (or parameter vector), the logistic-linear-function refers to $g(\mathbf{w}^\intercal \phi(\mathbf{x})) = 1/(1+e^{-\mathbf{w}^\intercal \phi(\mathbf{x})})$, where $\phi(\mathbf{x}) = [1, \mathbf{x}^\intercal]^\intercal$ is your augmented input vector, also denoted $\tilde{\mathbf{x}}$ in lecture notes, and logistic-quadratic-function refers to $g(\mathbf{w}^\intercal \phi(\mathbf{x})) = 1/(1+e^{-\mathbf{w}^\intercal \phi(\mathbf{x})})$, where $\phi(\mathbf{x}) = [1, x_1, x_2, x_1^2, x_1 x_2, x_2^2]^\intercal$.

Question 2 (50%)

Assume that scalar-real $y \in \mathbb{R}$ and two-dimensional real vector $\mathbf{x} \in \mathbb{R}^2$ are related to each other according to $y = c(\mathbf{x}, \mathbf{w}) + \varepsilon$, where $c(., \mathbf{w})$ is a cubic polynomial in $\mathbf{x}$ with coefficients $\mathbf{w}$, and ε is a random Gaussian scalar with mean zero and σ^2 variance ($\varepsilon \sim \mathcal{N}(0, \sigma^2)$).

Given a dataset $D = \{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\}$ with N samples of $(\mathbf{x}, y)$ pairs that are independent and identically distributed according to the model, derive two estimators for $\mathbf{w}$ using maximum-likelihood (ML) and maximum-a-posteriori (MAP) parameter estimation approaches. For the MAP estimator, assume that $\mathbf{w}$ has a zero-mean Gaussian prior with covariance matrix $\gamma \mathbf{I}$.

Having derived the estimator expressions, implement them in code and apply them to the dataset generated by the attached Python/Matlab script. Using the *training dataset* ($N_{train} = 100$), obtain the ML estimator and the MAP estimator for a variety of γ values ranging from 10^{-4} to 10^4 (span a log scale). Evaluate each *trained* model by calculating the mean squared error (MSE) between the y values in the *validation samples* ($N_{valid} = 1000$) and model estimates of these using $c(., \mathbf{w}_{train})$. How does your MAP-trained model perform on the validation set as γ is varied? How is the MAP estimate related to the ML estimate? Describe your experiments, visualize and quantify your analyses with data from these experiments (e.g. plot MSE on the validation dataset as a function of hyperparameter γ).

Note: Point split will be 25% for ML and 25% for MAP estimator results.