EECE5644 - Assignment 2

Dhruv Agarwal

October 2024

1 Question 1

1.1 Data Generation

Data is generated using the following method:

$$\mathbf{x}_i = r \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} + \mathbf{n}$$

where, r is chosen according to the label:

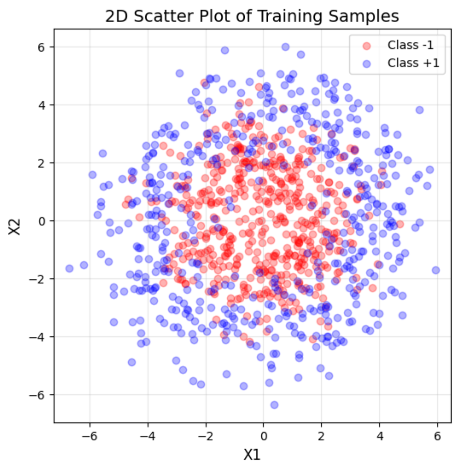
$$r = \begin{cases} 2, & \text{if } l = -1 \\ 4, & \text{if } l = +1 \end{cases}$$

l is chosen to be -1 with probability 0.5 and +1 with probability 0.5

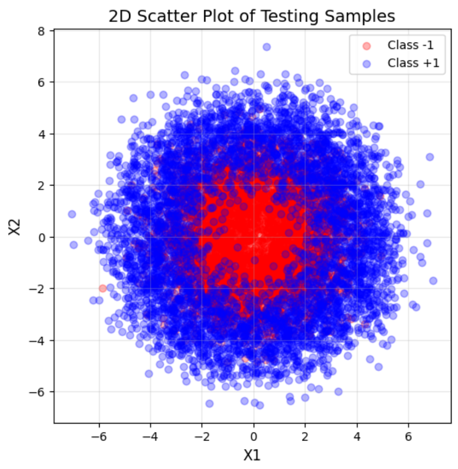
$$\theta \sim \text{Uniform}[-\pi, \pi]$$

$$\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$$

1000 i.i.d. training samples are generated:



10000 i.i.d. test samples are generated:



1.2 Support Vector Machine Model

- **Parameter Space:** Define the parameter search space:

$$C \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}, \quad \gamma \in \{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$$

- **Objective Function:** The support vector machine (SVM) minimizes the following objective function:

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i$$

  subject to:

$$y_i(\mathbf{w}\cdot\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \forall i$$

  where $\phi(\mathbf{x})$ is the mapping defined by the RBF kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$$

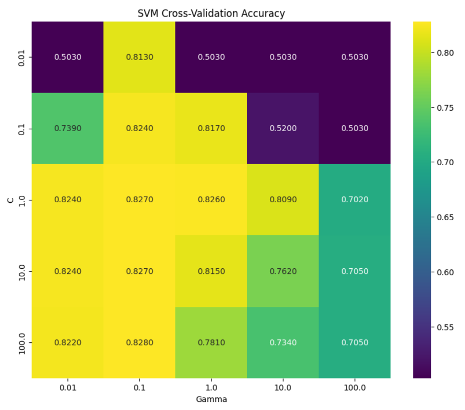
- **Cross-Validation:** Perform 10-fold cross-validation (CV) for all combinations of (C, γ):

$$\text{CV Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}}$$

  The best hyperparameters (C^*, γ^*) maximize the CV accuracy.

- The heatmap represents the cross-validation mean accuracy for each (C, γ) pair:

$$\text{Accuracy Matrix: } A_{ij} = \text{CV Accuracy for } C_i \text{ and } \gamma_j$$



```
Best Parameters: {'C': 100.0, 'gamma': 0.1}
Best Cross-Validation Accuracy: 0.8280
```

- **Plotting Decision Boundary** For visualization, the decision boundary is computed by evaluating the SVM's decision function over a grid of points:

  $$Z(x, y) = \text{sign}(\mathbf{w} \cdot \phi(\mathbf{x}) + b)$$

  The boundary is defined where $Z(x, y) = 0$.

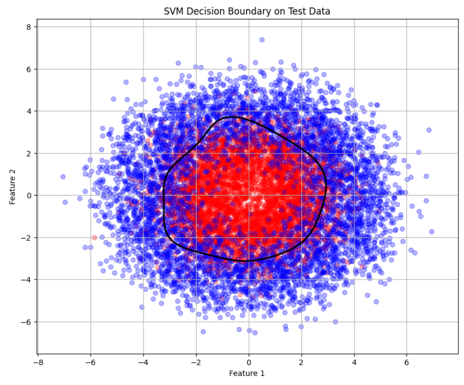
  The data points are plotted with:

  – Red for class -1 $(y = -1)$

  – Blue for class $+1$ $(y = +1)$

  Contours represent the decision boundary:

  $$\{(x, y) : Z(x, y) = 0\}$$



- **Prediction:** Predict the labels for the test set:

$$\hat{y}_i = \text{sign}(\mathbf{w} \cdot \phi(\mathbf{x}_i) + b)$$

  Test Accuracy: Compute the test accuracy:

$$\text{Test Accuracy} = \frac{\sum_{i=1}^{N} \mathbb{1}(\hat{y}_i = y_i)}{N}$$

  where $\mathbb{1}(\cdot)$ is the indicator function.
  Probability of Error: The probability of error is:

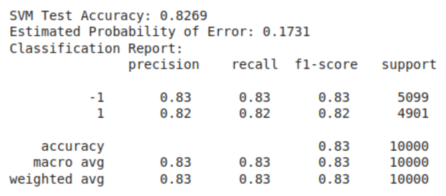
$$P_{\text{error}} = 1 - \text{Test Accuracy}$$

  Classification Report: Generate metrics such as precision, recall, and F1-score for each class:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$



```
SVM Test Accuracy: 0.8269
Estimated Probability of Error: 0.1731
Classification Report:
              precision    recall  f1-score   support

          -1       0.83      0.83      0.83      5099
           1       0.82      0.82      0.82      4901

    accuracy                           0.83     10000
   macro avg       0.83      0.83      0.83     10000
weighted avg       0.83      0.83      0.83     10000
```

1.3 Multi Layer Perceptron Model

- **Defining the MLP Model and Parameter search** We aim to optimize the hyperparameters of a Multi-Layer Perceptron (MLP) model using 10-fold cross-validation. Let:

$$\mathcal{H} = \{(5,),(10,),(15,),(20,)\}$$

  be the set of hidden layer sizes, and

$$\mathcal{A} = \{\text{logistic}, \text{relu}\}$$

  be the set of activation functions. We are not using quadratic activation functions because the in-built function for MLP in sklearn library only supports identity, relu, logistic and tanh. The regularization parameter is fixed as:

$$\alpha = 0.01.$$

The grid search evaluates each combination of parameters from $\mathcal{H}$ and $\mathcal{A}$ using 10-fold cross-validation.

For $i = 1, 2, \ldots, 10$ (cross-validation folds), define:

$$\text{Training Sets: } \{(X^i_{\text{train}}, Y^i_{\text{train}})\}, \quad \text{Validation Sets: } \{(X^i_{\text{val}}, Y^i_{\text{val}})\}.$$

The accuracy score for each parameter set (h, a) is computed as:

$$\text{Accuracy}(h, a) = \frac{1}{10} \sum_{i=1}^{10} \text{Accuracy}\left(f_{h,a}(X^i_{\text{val}}), Y^i_{\text{val}}\right),$$

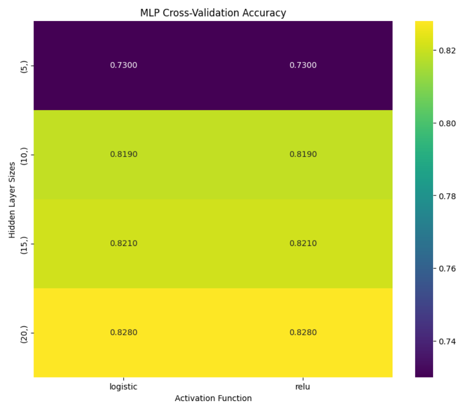
where $f_{h,a}$ represents the MLP model trained with parameters h and a.

The optimal parameters (h^*, a^*) are:

$$(h^*, a^*) = \underset{(h,a)}{\text{argmax}} \text{ Accuracy}(h, a).$$

- **Visualize Cross-Validation Results** A pivot table is created with:
  - Rows indexed by $\mathcal{H}$ (hidden layer sizes).
  - Columns indexed by $\mathcal{A}$ (activation functions).
  - Values given by $\text{Accuracy}(h, a)$.

  A heatmap represents these results.

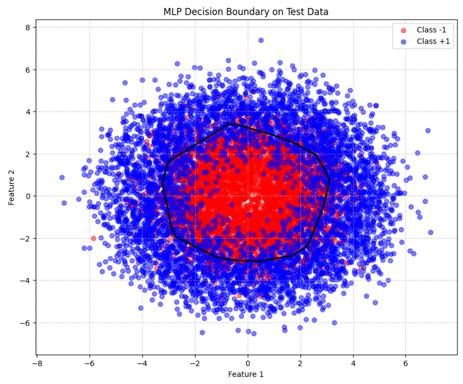


```
Best Parameters: {'activation': 'logistic', 'alpha': 0.01, 'hidden_layer_sizes': (20,)}
Best Cross-Validation Accuracy: 0.8280
```

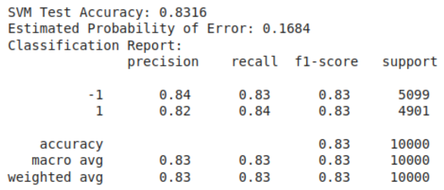
- **Decision Boundary Visualization** To visualize the decision boundary:
    1. Create a grid of points (x', y') where:
    $$x', y' \in [\min(X_1) - 1, \max(X_1) + 1] \times [\min(X_2) - 1, \max(X_2) + 1].$$
    2. Predict the class $Z(x', y')$ for each grid point using f_{h^*,a^*}.
    3. Plot the boundary where:
    $$Z(x', y') = 0 \quad \text{(decision boundary)}.$$

    Overlay the scatter plot of the data points:

    - $X_{\text{test}}^{(-1)}$ (class -1) in red.
    - $X_{\text{test}}^{(+1)}$ (class $+1$) in blue.



- **Testing the Best Model** After selecting the best model f_{h^*,a^*}, test it on $(X_{\text{test}}, Y_{\text{test}})$:
$$\text{Test Accuracy} = \text{Accuracy}\left(f_{h^*,a^*}(X_{\text{test}}), Y_{\text{test}}\right).$$
The estimated probability of error is:
$$\text{Error Probability} = 1 - \text{Test Accuracy}.$$
Additionally, generate the classification report:
$$\text{Precision, Recall, F1-Score} \quad \text{for each class.}$$



```
SVM Test Accuracy: 0.8316
Estimated Probability of Error: 0.1684
Classification Report:
              precision    recall  f1-score   support

          -1       0.84      0.83      0.83      5099
           1       0.82      0.84      0.83      4901

    accuracy                           0.83     10000
   macro avg       0.83      0.83      0.83     10000
weighted avg       0.83      0.83      0.83     10000
```

2 Question 2

2.1 Image Feature Vector Calculation

- **Image Dimensions** Let the image I have dimensions $H \times W$ (height $\times$ width) with 3 color channels (red, green, blue). The pixel intensity values are denoted as $I(r, c)$ for row $r \in \{0, \dots, H-1\}$ and column $c \in \{0, \dots, W-1\}$.
- **Pixel Position Indices**

$$\text{row\_indices}(r, c) = r, \quad \text{col\_indices}(r, c) = c \quad \forall r, c$$

- **Color Normalization**

$$\text{red\_values}(r, c) = \frac{I(r, c, \text{Red})}{255}, \quad \text{green\_values}(r, c) = \frac{I(r, c, \text{Green})}{255}, \quad \text{blue\_values}(r, c) = \frac{I(r, c, \text{Blue})}{255}$$

- **Feature Vector** For each pixel, the feature vector is:

$$f(r, c) = \begin{bmatrix} r \\ c \\ \text{red\_values}(r, c) \\ \text{green\_values}(r, c) \\ \text{blue\_values}(r, c) \end{bmatrix}$$

The feature vectors for all pixels are stacked together:

$$\text{features} = \{f(r, c)\}_{r=0, c=0}^{H-1, W-1}$$

- **Normalization** Features are normalized using Min-Max scaling:

$$\text{features\_normalized} = \frac{\text{features} - \min(\text{features})}{\max(\text{features}) - \min(\text{features})}$$

2.2 K-Fold Training of Gaussian Mixture Model (GMM)

- **K-Fold Splitting** Divide the dataset into K subsets (folds) $\{D_1, D_2, \dots, D_K\}$.
- **Log-Likelihood Evaluation** For $n_{\text{components}} \in \{1, \dots, 10\}$, train a Gaussian Mixture Model $GMM(n_{\text{components}})$ and compute the average log-likelihood:

$$\text{Log\_Likelihood}_k = \frac{1}{|D_k|} \sum_{x \in D_k} \log P(x \mid GMM(n_{\text{components}}))$$

The average log-likelihood across folds is:

$$\text{Avg\_Log\_Likelihood}(n_{\text{components}}) = \frac{1}{K} \sum_{k=1}^{K} \text{Log\_Likelihood}_k$$

- **Model Selection** Choose the number of components that maximizes the average log-likelihood:

$$n_{\text{best}} = \arg \max_{n_{\text{components}}} \text{Avg\_Log\_Likelihood}(n_{\text{components}})$$

- **Fit the Final GMM** Train $GMM(n_{\text{best}})$ on the full normalized feature set:

$$GMM_{\text{final}} = GMM(n_{\text{best}})$$

- **Label Assignment** Predict cluster labels for the pixels:

$$\text{labels} = \{GMM_{\text{final}}.\text{predict}(f(r,c))\}_{r=0,c=0}^{H-1,W-1}$$

- **Label Reshaping** Reshape the label vector into an image:

$$\text{labels\_image}(r,c) = \text{labels}[(r \cdot W) + c]$$

2.3 Grayscale Image Segmentation

- **Normalize Labels to Grayscale** Map cluster labels to grayscale values:

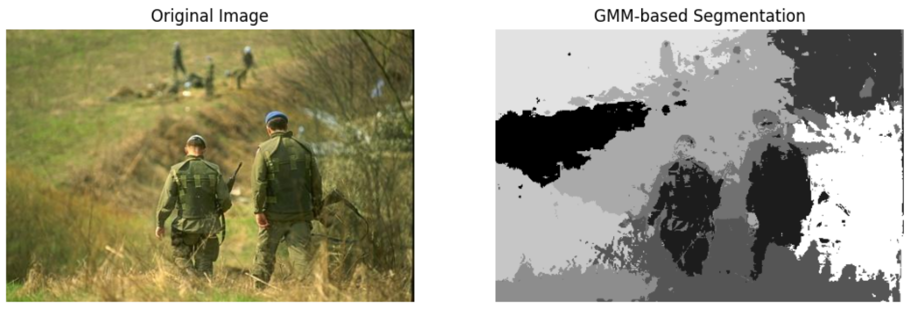
$$\text{label\_image\_gray}(r,c) = 255 \cdot \frac{\text{labels\_image}(r,c) - \min(\text{labels\_image})}{\max(\text{labels\_image}) - \min(\text{labels\_image})}$$

- **Visualization** Display the original image:

$$\text{Original Image} = I$$

Display the grayscale segmentation:

$$\text{Segmented Image} = \text{label\_image\_gray}$$



2.4 Color Image Segmentation

- **Unique Labels and Palette Generation** The unique labels in the segmented image are:

$$L = \{\ell_1, \ell_2, \dots, \ell_N\}, \quad N = |L|$$

Generate a random color palette for each label:

$$\text{palette}(\ell_i) = [R_i, G_i, B_i] \in \{0, 1, \dots, 255\}^3, \quad i = 1, \dots, N$$

- **Constructing the Color Image** Initialize a color image of size $H \times W$:

$$\text{label\_image\_color}(r, c) = [0, 0, 0], \quad \forall (r, c) \in \{0, \dots, H-1\} \times \{0, \dots, W-1\}$$

Assign colors based on labels:

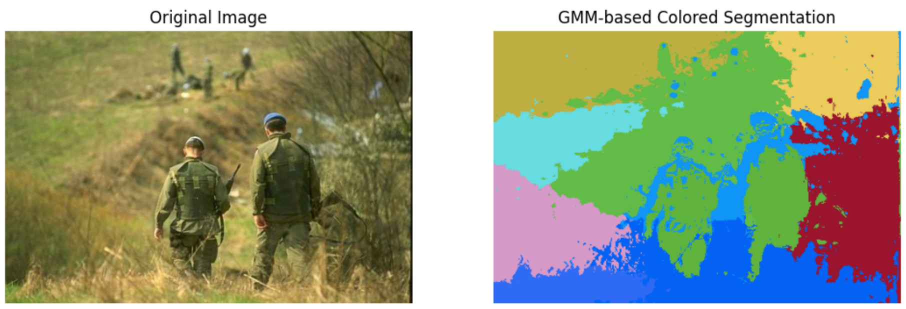
$$\text{label\_image\_color}(r, c) = \text{palette}(\text{labels\_image}(r, c)), \quad \forall (r, c)$$

- **Visualization** Display the original image:

$$\text{Original Image} = I$$

Display the color-labeled segmentation:

$$\text{Segmented Image (Color)} = \text{label\_image\_color}$$



Link to the GitHub Folder or copy this into your web browser:
https://github.com/Dhruv-2020EE30592/EECE-5644/tree/main/Assignment-4