ASSIGNMENT 2

CS5691 Pattern Recognition and Machine Learning

# CS5691 Assignment 2


Team Members:

| | |
|---|---|
| BE17B007 | N Sowmya Manojna |
| PH17B010 | Thakkar Riya Anandbhai |
| PH17B011 | Chaithanya Krishna Moorthy |

Indian Institute of Technology, Madras

# Contents

# 1 Dataset 1A

## 1.1 K-nearest Neighbors Classifier

The K Nearest Neighbour is a statistically non-parametric model that can be used for regression as well as for classification. It assumes that similar things exist in close proximity. Crucial steps in a K-Nearest Neighbour classifier are:

- A distance metric is first specified, the most commonly used metric is the euclidean distance:

$$d = ||\vec{x_1} - \vec{x_2}|| \tag{1}$$

  where $||.||$ denotes the norm function. Other commonly used distance metrics are the Manhattan distance and cosine similarity. For our application, euclidean distance is used.

- Using the specified distance metric, the distance between the test instance and each training example is evaluated.

- The class label that occurs most frequently amongst the nearest k training examples is assigned to the test instance

Advantages of KNN are:

- KNN does not require a training period, it just stores the training dataset and learns from it at the time of making a prediction, hence it is generally much faster than other classification algorithm.

- Since the algorithm does not require prior training, new data points can be added seamlessly.

- Easy to implement,the number of parameters are just 2- k and the distance metric to be used

Disadvantages of KNN are:

- Computationally expensive for large data sets or high number of features, since the distance is evaluated between test point and all the points in the training data set.

- Sensitive to noisy data and outliers. Generally, increasing the value of k reduces the effect of noise.

### 1.1.1 Pre-processing

The data set 1a has 4 unique class labels - 1.0,0.0,2.0,3.0 as shown in the figure. Number of examples corresponding to each class label is 200. The train data set is of dimension (800,3) while the dev data set is of dimension (120,3).The 3rd column in both data sets is the class label, while the first two columns are the real valued feature vectors- $x_1 and x_2$

1. There are no null values in the data sets.

2. The rows of dev data set are shuffled and further split into cross-validation and test data in the ratio of 70:30

3. Range of $x_1$ is (-11,11) and range of $x_2$ is (-12,7). Since the ranges are almost similar, no feature scaling is required.

### 1.1.2 Model performance for varying values of k

The model was evaluated for k values = 1,7,15. We find that irrespective of the value of hyper-parameter k, the model obtained an accuracy of 1 over training data, cross-validation data as well as the test data. Since model performance is best irrespective of k, the accuracy table, confusion matrix and decision boundary plot are all evaluated using k=1 as to minimize the run time.
The accuracy table and confusion matrix are:

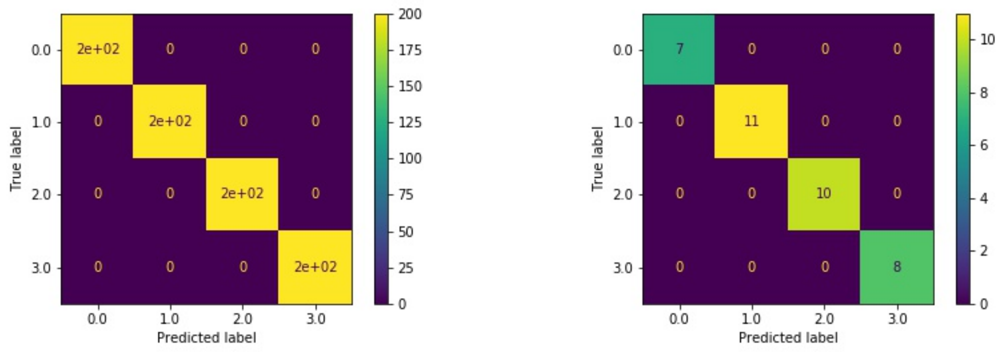



**Figure 1:** Confusion matrix for k=1, Train and Test data set on left and right respectively

### 1.1.3 Decision region plot with training data superposed

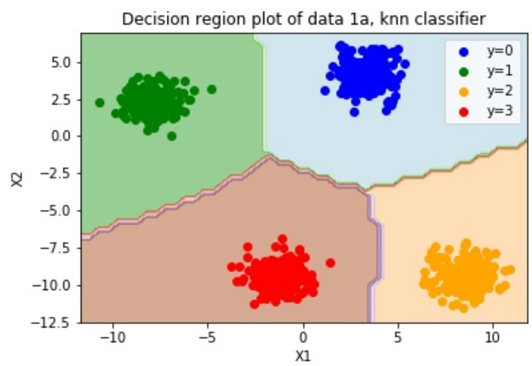



**Figure 2:** Decision region superimposed with training data set

The decision boundary obtained (with k=1) is linear in form.

## 1.2 Naive-Bayes classifier with Gaussian distribution for each class

The Bayes Classifiers are probabilistic classifiers based on the Bayes theorem:

$$p(y_i/\vec{x}) = \frac{p(\vec{x}/y_i) * p(y_i)}{p(\vec{x})} \tag{2}$$

Here,
$p(y_i)$ : prior probability for $y = y_i$.
$p(\vec{x}/y_i)$ : Class conditional probability density function or class conditional likelihood function.
$p(y_i/\vec{x})$ : Posterior probability for $y = y_i$ given $\vec{x}$.
$p(\vec{x})$ : Evidence or normalization factor.

Equation 2 can be re-written as:

$$p(y_i/\vec{x}) = \frac{p(\vec{x}/y_i) * p(y_i)}{\sum_i p(\vec{x}/y_i) * p(y_i)} \tag{3}$$

Hence, the probability that $\vec{x}$ belongs to the class $y_i$ is $\propto p(\vec{x}/y_i) * p(y_i)$.

Step-wise approach:

- $p(y_i)$ is calculated from the train data set, for data set 1a, we find that all the classes have equal prior probability.

- The probability $p(\vec{x}/y_i)$ can be calculated by various parametric and non-parametric means. For data set 1a, we use parametric means as described later.
- $p(y_i/\vec{x})$ is calculated for all the classes using equation 3.
- The class label with maximum posterior probability is chosen as the class label for $\vec{x}$.

For the discussion that follows for data set 1a, we assume that $p(\vec{x}/y_i)$ is given by a gaussian distribution:

$$p(\vec{x}/y_i) = \frac{exp[-(\vec{x} - \vec{\mu_i})^t * C_i^{-1} * (\vec{x} - \vec{\mu_i})/2]}{(2\pi)^{d/2} * |C_i|^{1/2}} \tag{4}$$

In the above equation:

- $\mu_i$ is the mean corresponding to examples in the class $y_i$, its dimension is d*1, where d is the number of features. Hence, if there are k classes, number of parameters to be estimated for mean = k*d
- $C_i$ is the d*d co-variance matrix corresponding to the class $y_i$. Since it is symmetric, number of parameters to be calculated per class = $\frac{d(d+1)}{2}$. Total parameters for co-variance = $\frac{k*d(d+1)}{2}$

### 1.2.1 Case a : Same Covariance Matrix ($\sigma^2 I$)

To reduce the number of parameters to be calculated, we assume that

$$C_i = C_j = \sigma^2 I \tag{5}$$

$$\sigma^2 = \frac{\sum_{i=1}^{d} \sum_{k=1}^{K} \sigma_{ik}^2}{K * d} \tag{6}$$

Where, K is the number of classes (4 for data set 1a) and d is the dimension of feature vector (2 for data set 1a) Substituting (6) in equation 4, $p(\vec{x}/y_i)$ is calculated and used for predicting the class labels. This is also called the naive-bayes classifier since we assume the features to be conditionally independent. The decision boundary obtained is linear, while the level curves are circles.

### 1.2.2 Case b : Same Co-variance Matrix ($C$)

The covariance matrix C is calculated as:

$$C = \frac{\sum_k^K C_k}{K} \tag{7}$$

Where $C_k$ is the co-variance matrix corresponding to the kth class.

With this assumption, the decision boundary is linear, while the level curves are ellipses with equal length of principal axes, proportional to the eigen-vectors of C.

### 1.2.3 Case c : Different Co-variance Matrix

In this case, the decision surfaces are hyper-quadrics.

### 1.2.4 Accuracy table and Confusion Matrix

We obtain that irrespective of our assumption of the co-variance matrices, the accuracy over train, validation and test set is 1.

# 2 Dataset 1B

## 2.1 K-nearest Neighbors Classifier

## 2.2 Bayes Classifier, GMM, full covariance

### 2.2.1 Equations

Th initialization is done as follows for each class:

- Cluster initialization is using `kmeans` clustering.
- The relative number of points in each cluster $N_q$ and weightage $w_q$ for each cluster is calculated.
- The responsibility $\gamma_{n,q}$ is then calculated, followed by mean $\mu_q$ and covariance $C_q$ is calculated.

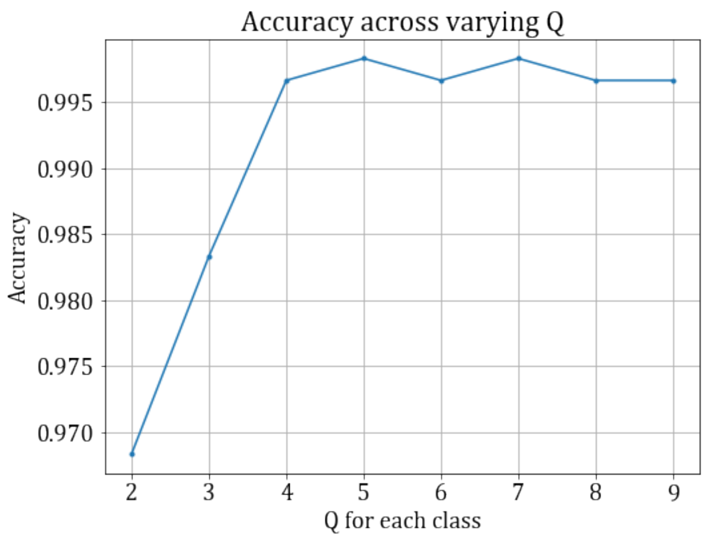
The parameters are then updated sequentially through the:

- Expectation-step: $\gamma_{n,q}$ is updated.
- Maximization-step: $\mu_q$, $C_q$, $N_q$ and $w_q$ are updated.

The stopping criterion used is $\Delta(\text{likelihood}) < \texttt{tol}$. The `tol` we considered is $10^{-5}$.

### 2.2.2 Training and Validation Accuracy

The training and validation accuracies obtained for varying $q_i$ for each class is as follows:



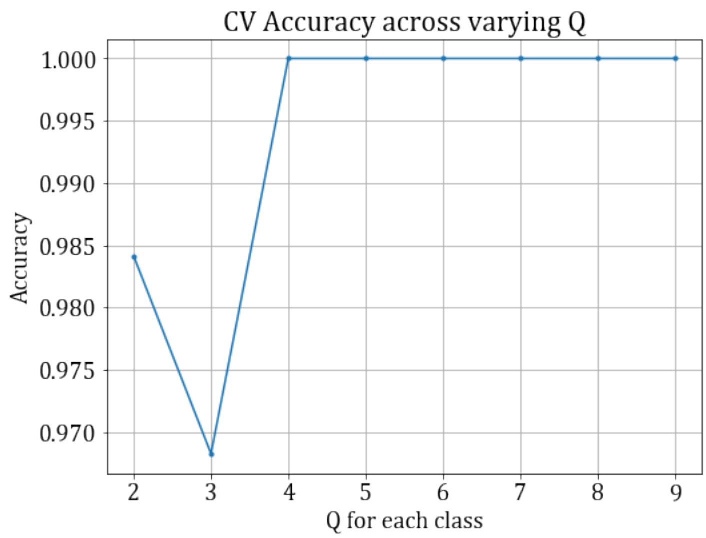




**Figure 3:** Training and Validation accuracy across $q_i$, on the left and right respectively

### 2.2.3 Testing Accuracy

The testing accuracy obtained for varying $q_i$ for each class is as follows:

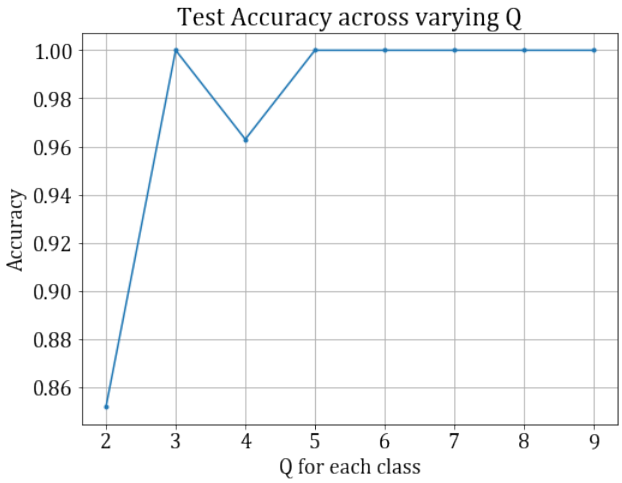



**Figure 4:** Testing accuracy across $q_i$

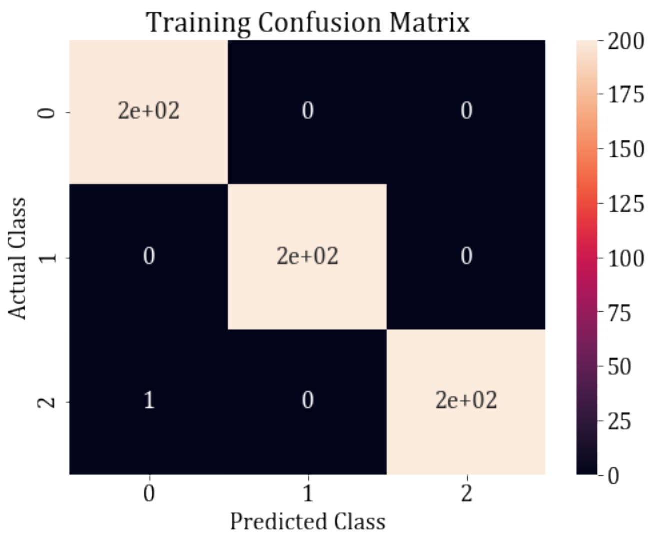
#### 2.2.4 Best Model

Based on the accuracies obtained on the training, validation and test dataset, the best $q_i$ for the three classes has been chosen as 5. The accuracies obtained in tabular format is as follows:

| Number of Clusters/Class (Q) | Train Accuracy | Validation Accuracy | Test Accuracy |
|---|---|---|---|
| 2 | 0.968333 | 0.952381 | 0.925926 |
| 3 | 0.983333 | 0.968254 | 1.000000 |
| 4 | 0.996667 | 0.984127 | 1.000000 |
| 5 | 0.998333 | 1.000000 | 1.000000 |
| 6 | 0.996667 | 1.000000 | 1.000000 |
| 7 | 0.998333 | 1.000000 | 1.000000 |
| 8 | 0.996667 | 1.000000 | 1.000000 |
| 9 | 0.996667 | 1.000000 | 1.000000 |

**Table 1:** Variation of accuracy across hyperparameter values on the training, validation and test set using the GMM model with full covariance matrix on Dataset 1B

The confusion matrix obtained for the model with $q_i = 5$ are as follows:



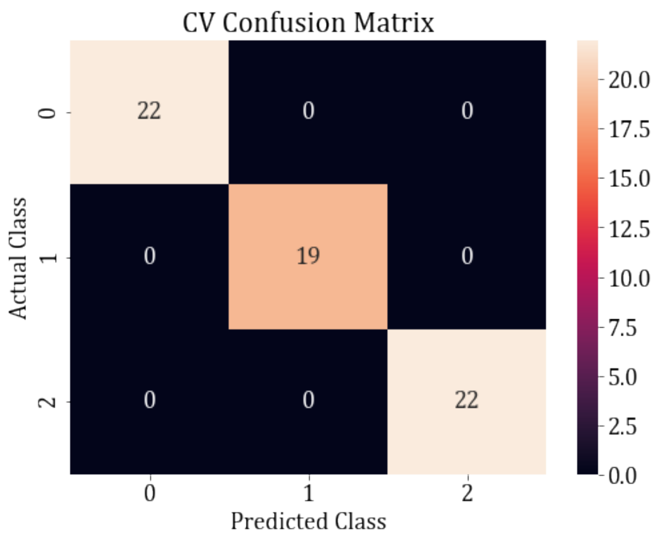




**Figure 5:** Training and Validation confusion matrices for the best model with $q_i = 5$, on the left and right respectively

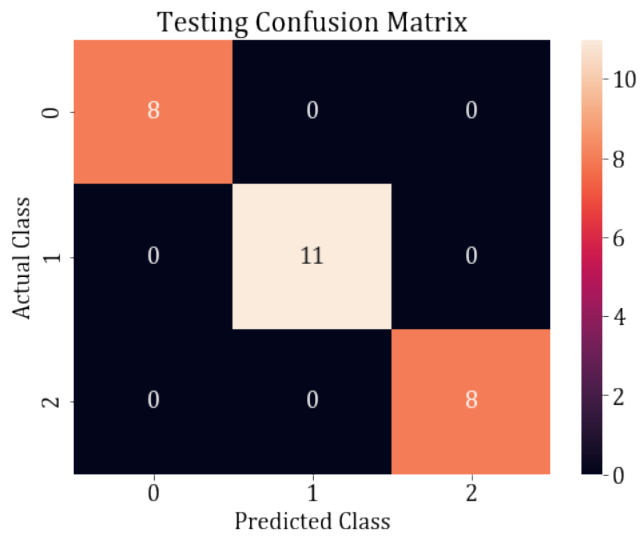



**Figure 6:** Testing confusion matrix for the best model with $q_i = 5$.

#### 2.2.5 Contour Maps and Decision Surfaces

The contour maps and decision surfaces obtained, with $q_i = 5$ are as follows:

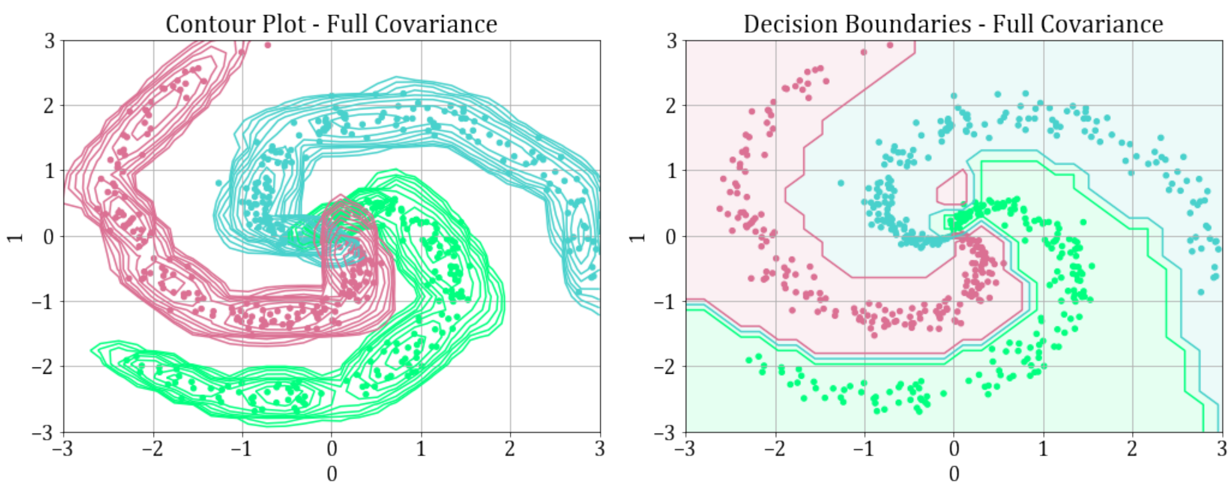



**Figure 7:** Contour Maps, Decision Surfaces obtained for $q_i = 5$, on the left and right respectively.

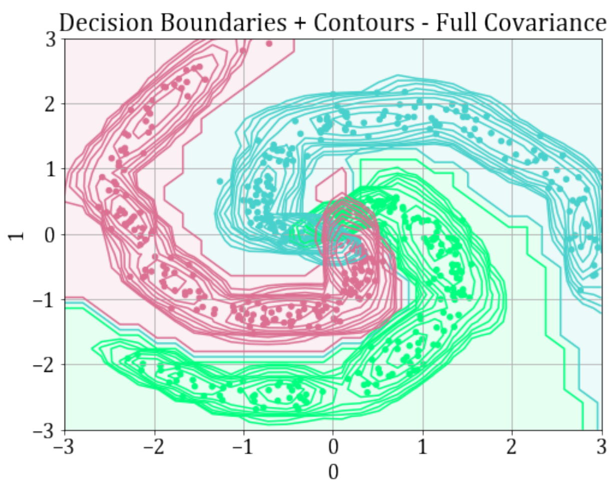



**Figure 8:** Overlap plot of the decision surface and contours.

## 2.3 Bayes Classifier, GMM, diagonal covariance

### 2.3.1 Training and Validation accuracy

Gaussian multi-modal training function (with threshold of the increment in total log-likelihood functions as 0.01 since there was no significant improvement for a smaller threshold than this) with diagonal covariance matrix over the hyperparameter values of the number of gaussian components Q = 2,3,4,5,6,7,8,9 to estimate the parameters - $\mu_q$, $C_q$, $N_q$ and $w_q$ for each gaussian component - and predict the classes of the training data (train.csv) and cross-validation (70% of dev.csv), we get the table 2

| Hyperparameter Value (Q) | Accuracy on CV data | Accuracy on Training data |
|---|---|---|
| 2 | 0.873 | 0.9166 |
| 3 | 0.920 | 0.976 |
| 4 | 0.968 | 0.9966 |
| 5 | 0.984 | 1.0 |
| 6 | 0.984 | 0.986 |
| 7 | 0.984 | 0.991 |
| 8 | 0.984 | 0.9916 |
| 9 | 0.984 | 0.9916 |

**Table 2:** Variation of Accuracy across Hyperparameter values on the validation data using the GMM model with diagonal covariance matrix on Dataset 1B

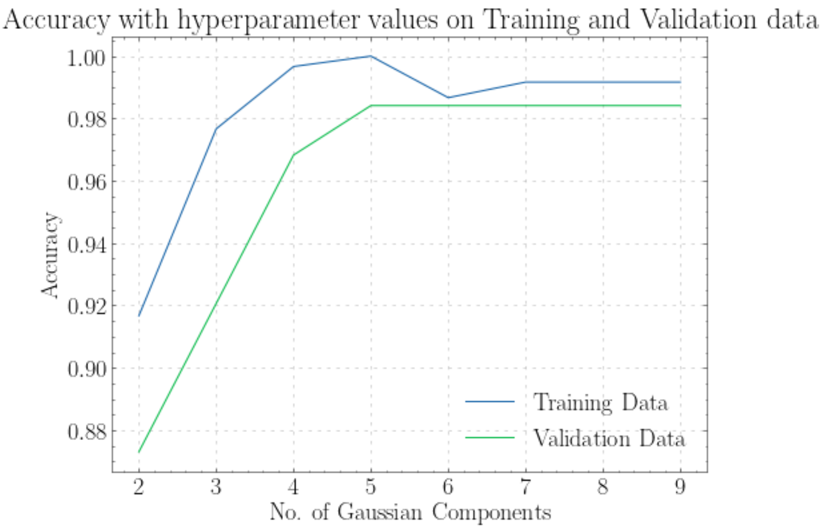



**Figure 9:** Plot of Hyperparameter value Vs Accuracy using the GMM model with diagonal covariance matrix on Dataset 1B

### 2.3.2 Best model output

As we can see in the tables and figure **??**, the best accuracy is when the number of Gaussian components is 5. Using the parameters of the model for 5 gaussian components and predicting for the test dataset (30% of dev.csv), the accuracy obtained was **1.0**.
The confusion matrices for the training and test datasets using the best model are tables 4 and **??**.

| | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 200 | 0 | 0 |
| 1 | 0 | 200 | 0 |
| 2 | 0 | 0 | 200 |

**Table 3:** Confusion Matrix for training data 1B

The decision region plot for the best model is figure 10

| | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 9 | 0 | 0 |
| 1 | 0 | 10 | 0 |
| 2 | 0 | 0 | 8 |

**Table 4:** Confusion Matrix for test data 1B

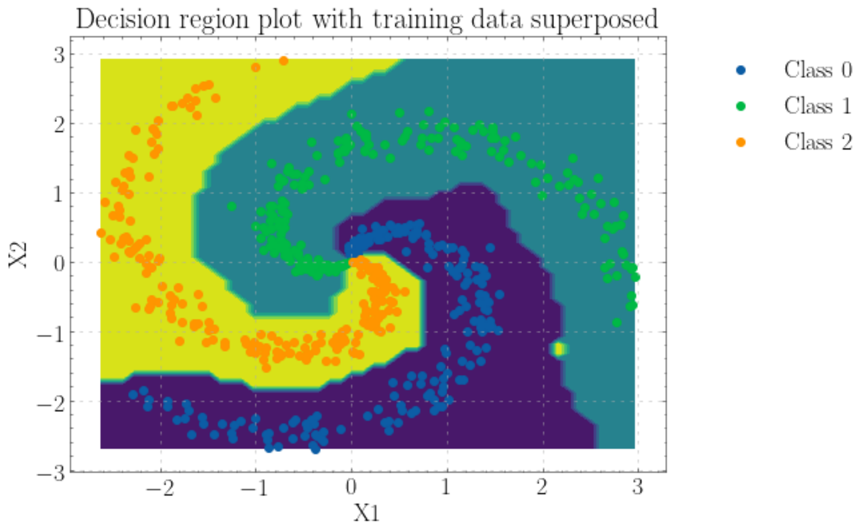



**Figure 10:** Decision region plot for Bayesian GMM model using diagonal covariance matrix and 5 gaussian components on dataset 1B

## 2.4 Bayes Classifier, KNN

# 3 Dataset 2A

## 3.1 Bayes Classifier, GMM, full covariance

## 3.2 Bayes Classifier, GMM, diagonal covariance

### 3.2.1 Training and Validation Accuracy

The accuracy obtained on training the data 2A on GMM model with diagonal covariance matrix is as in table 6. The plot of the same is in figure 11. The tolerence used was 1e-3.

| Hyperparameter Value | Training Accuracy | Validation Accuracy |
|---|---|---|
| 2 | 0.509 | 0.350 |
| 3 | 0.525 | 0.404 |
| 4 | 0.574 | 0.436 |
| 5 | 0.627 | 0.420 |
| 6 | 0.6491 | 0.418 |
| 7 | 0.663 | 0.440 |
| 8 | 0.689 | 0.371 |
| 9 | 0.692 | 0.413 |
| 10 | 0.7184 | 0.4272 |
| 11 | 0.735 | 0.396 |
| 12 | 0.754 | 0.393 |
| 13 | 0.770 | 0.434 |
| 14 | 0.783 | 0.388 |

**Table 6:** Table of Hyperparameter value Vs Accuracy for the Validation data using the GMM model with diagonal covariance matrix on Dataset 2A


**Figure 11:** Accuracy for training and validation set for 2A

## 3.3 Best model on test data

The highest accuracy on validation data set is for 7 gaussian components. Applying this model to predict the test data, we get an accuracy of **0.37**. The confusion matrices for this model on training and test data are tables 8 and **??**.

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 147 | 8 | 21 | 22 | 35 |
| 1 | 7 | 156 | 7 | 13 | 14 |
| 2 | 39 | 5 | 165 | 30 | 45 |
| 3 | 25 | 7 | 20 | 190 | 24 |
| 4 | 33 | 6 | 36 | 32 | 143 |

**Table 7:** Confusion Matrix for training data 2A

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 37 | 4 | 11 | 12 | 7 |
| 1 | 5 | 25 | 6 | 3 | 4 |
| 2 | 12 | 10 | 27 | 19 | 20 |
| 3 | 9 | 8 | 12 | 40 | 21 |
| 4 | 10 | 5 | 15 | 8 | 23 |

**Table 8:** Confusion Matrix for test data 2A

# 4 Dataset 2B

## 4.1 Bayes Classifier, GMM, full covariance

## 4.2 Bayes Classifier, GMM, diagonal covariance