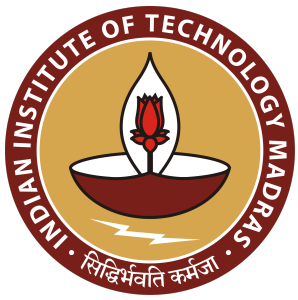
ASSIGNMENT 2

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 - $\{0.0, 1.0, 2.0, 3.0\}$ as shown in Figure 1. 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 third 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.

- There are no null values in the data sets.
- The rows of dev data set are shuffled and further split into cross-validation and test data in the ratio of 70:30
- 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:

k-value	Train accuracy	Cross-validation accuracy	Test Accuracy
1	1.00	1.00	1.00
7	1.00	1.00	1.00
15	1.00	1.00	1.00

Table 1: Accuracy table for data set 1a- knn classifier

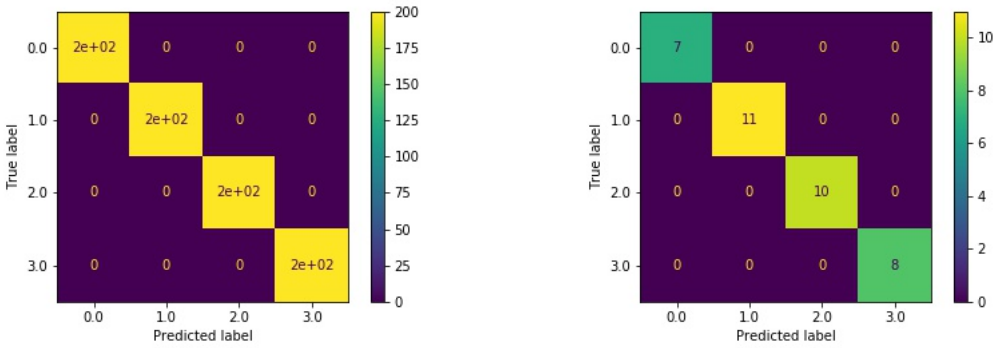


Figure 1: Confusion matrix for k=1, Train and Test data set on left and right respectively

1.1.3 Decision region plot with training data superposed

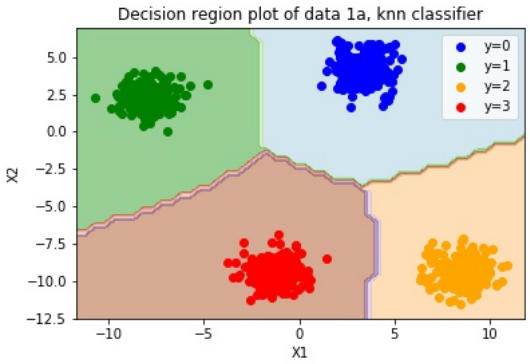


Figure 2: Decision region superimposed with training data set

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:

- μ_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 k^{th} 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.

Assumption	Train Accuracy	Validation Accuracy	Test Accuracy
$C_i = C_j = \sigma^2 I$	1.00	1.00	1.00
$C_i = C_j = C$	1.00	1.00	1.00
$C_i \neq C_j$	1.00	1.00	1.00

Table 2: Accuracy table for data set 1a: Bayes Classifier

2 Dataset 1B

2.1 K Nearest Neighbour Classifier

Similar to section **??**, K Nearest Neighbour classifier is used to predict class labels for data set 2A.
The data set 1B has 3 unique class labels - [0.0,1.0,2.0] as shown in the figure, it is also non-linearly separable. The train data set is of dimension (800,3) while the dev data set is of dimension (90,3). Similar preprocessing steps are performed as in section **??**

2.1.1 Model performance for varying values of K

Unlike data set 1A, we observe that the accuracy over validation set decreases on increasing the value of hyper-parameter k. This happens because the data set 1b is non-linear, a higher k value includes points from other class labels resulting in misjudgement.

The accuracy table is as follows:
Hence, the best configuration of the model exists for k=1.The confusion matrix in this case is:

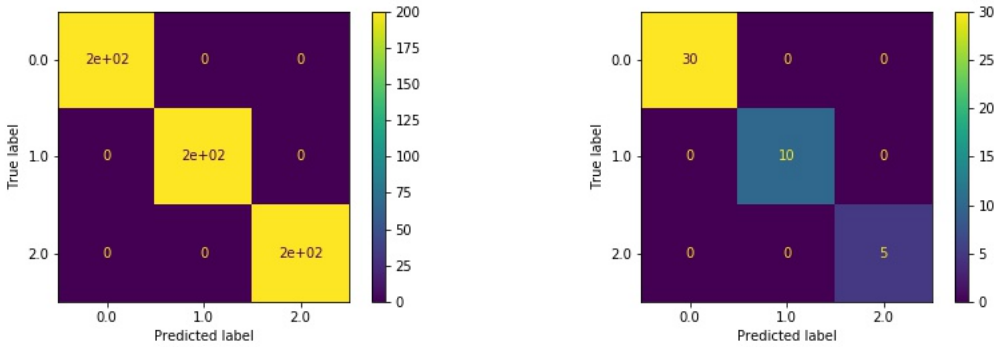


Figure 3: Confusion matrix for k=1, for Train and Test data on left and right respectively

2.1.2 Decision Boundary plot

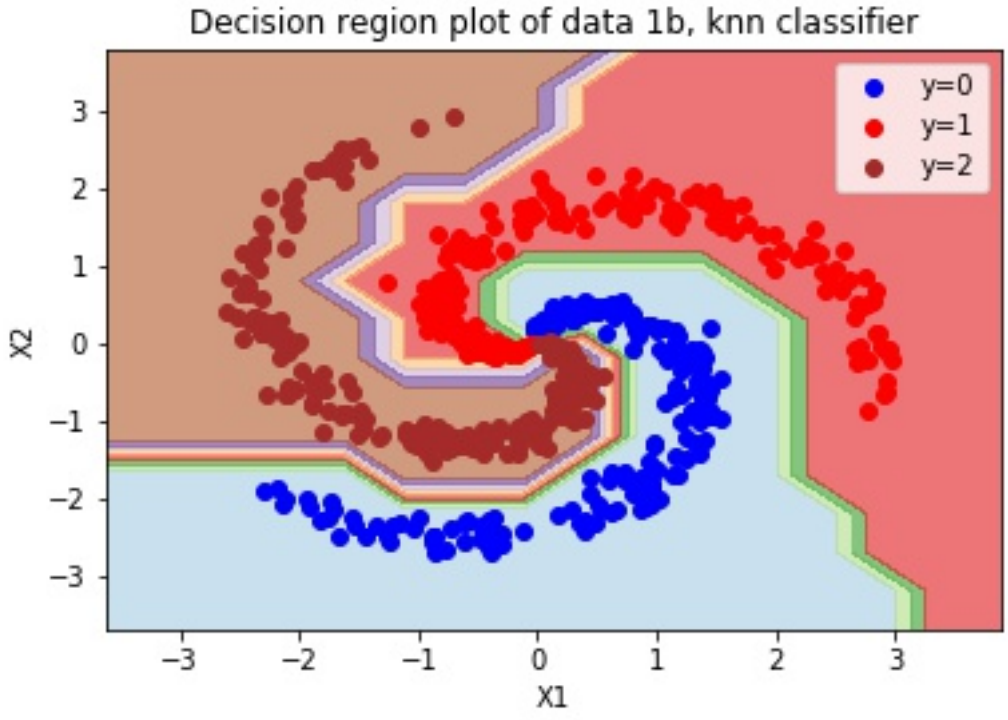


Figure 4: Decision boundary plot for k=1

The decision boundary obtained is non-linear in form and not smooth.

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 μ_q and covariance C_q is calculated.

The parameters are then updated sequentially through the:

- Expectation-step: $\gamma_{n,q}$ is updated.
- Maximization-step: μ_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:

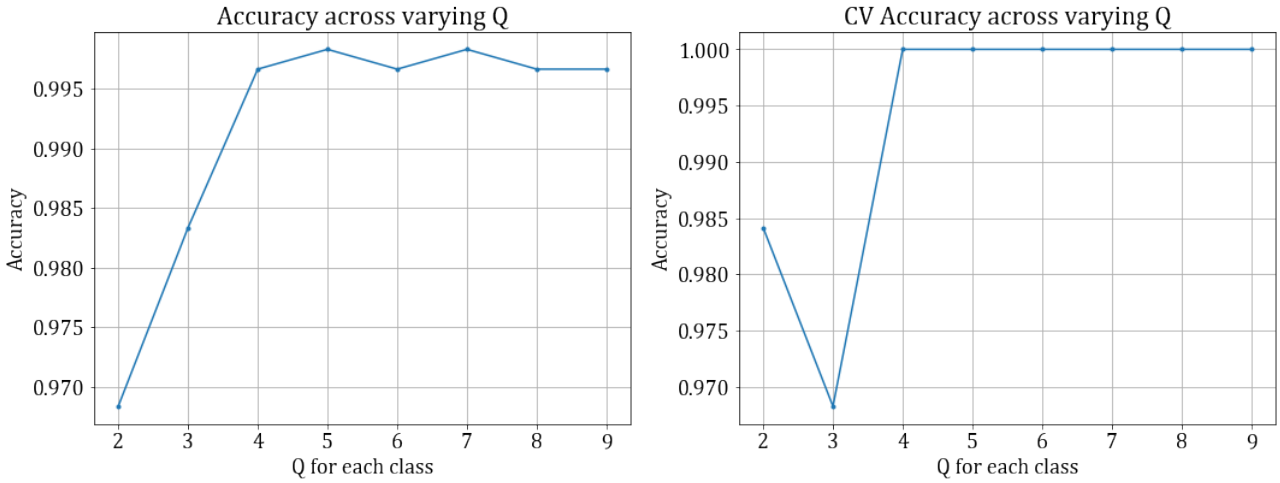


Figure 5: Training and Validation accuracy across q_i, on the left and right respectively, using a GMM model with full covariance matrix.

2.2.3 Testing Accuracy

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

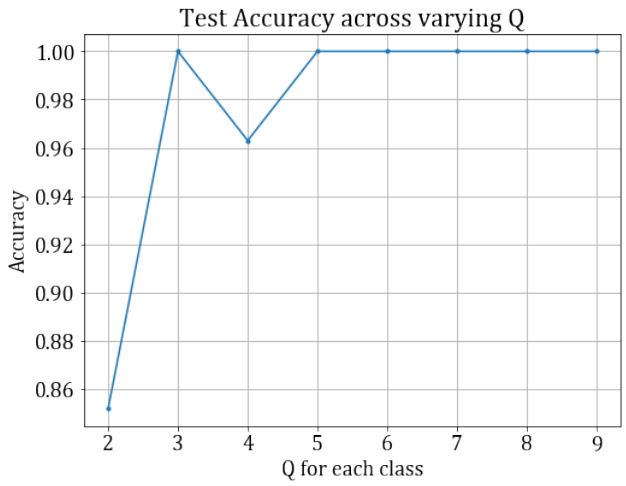


Figure 6: Testing accuracy across q_i, using a GMM model with full covariance matrix.

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:

# 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 3: Variation of accuracy across hyperparameter values on the training, validation and test set using full covariance matrix GMM model on Dataset 1B. The row corresponding to the best model has been highlighted.

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

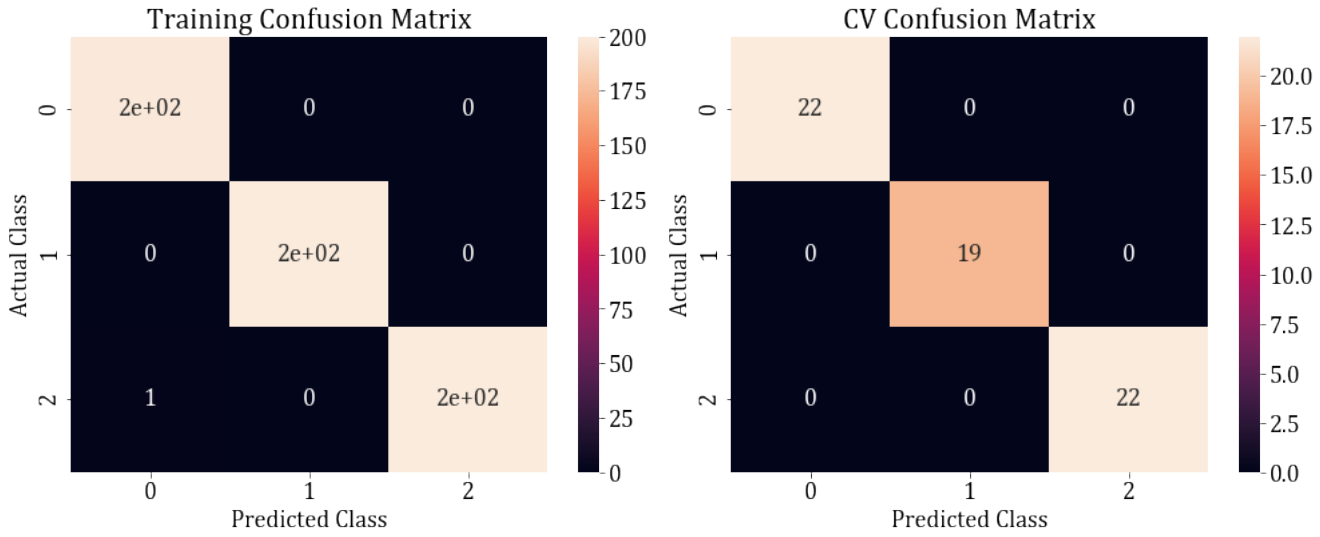


Figure 7: Confusion matrices corresponding to training and validation data, with $q_i = 5$, on the left and right respectively, using GMM model with full covariance.

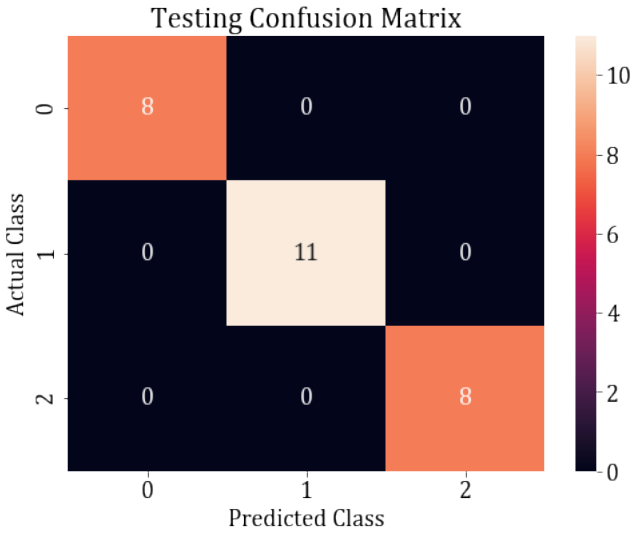


Figure 8: Confusion matrix corresponding to the testing data, with $q_i = 5$, using GMM model with full covariance.

2.2.5 Contour Maps and Decision Surfaces

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

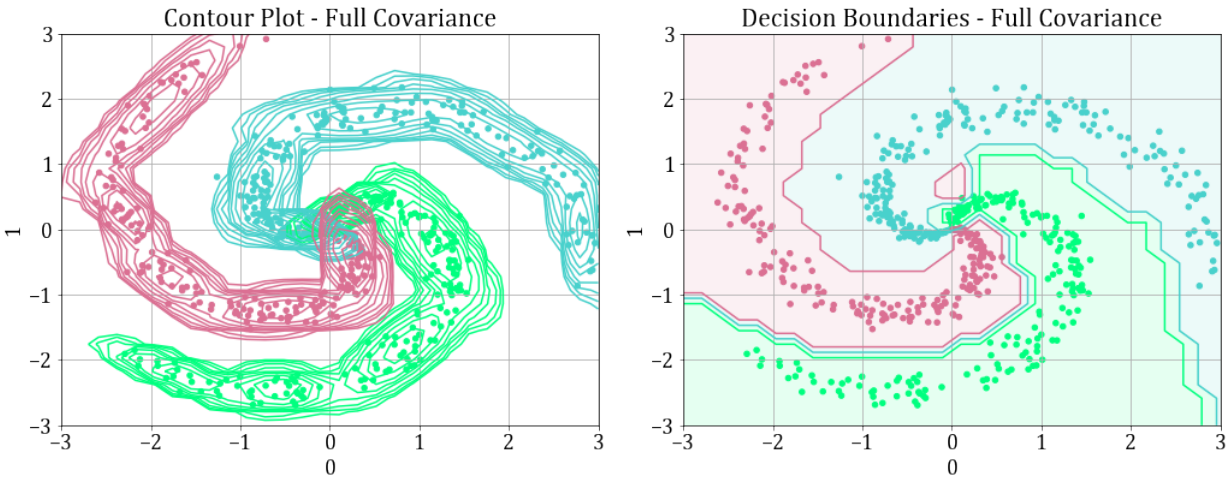


Figure 9: Contour Maps, Decision Surfaces obtained for $q_i = 5$, on the left and right respectively.

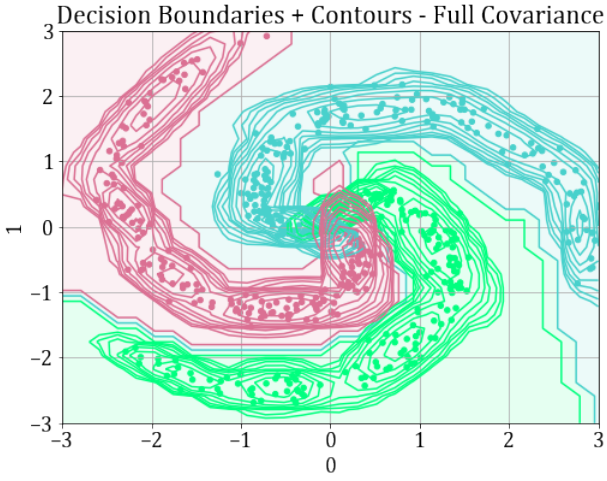


Figure 10: Overlap plot of the decision surface and contours.

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) 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 - μ_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 4

# Clusters/Class (Q)	Validation Accuracy	Training Accuracy
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 4: Variation of accuracy across hyperparameter values on the validation data using the GMM model with diagonal covariance matrix on Dataset 1B. The row corresponding to the best model has been highlighted.

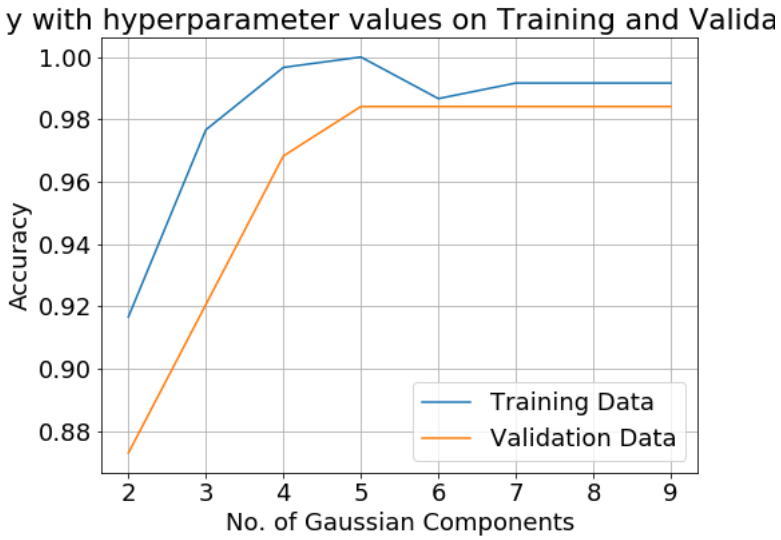


Figure 11: Training and Validation accuracy across q_i, using the GMM model with diagonal covariance matrix on Dataset 1B

2.3.2 Best model output

As we can see in the tables and figure 11, 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 is figure 12

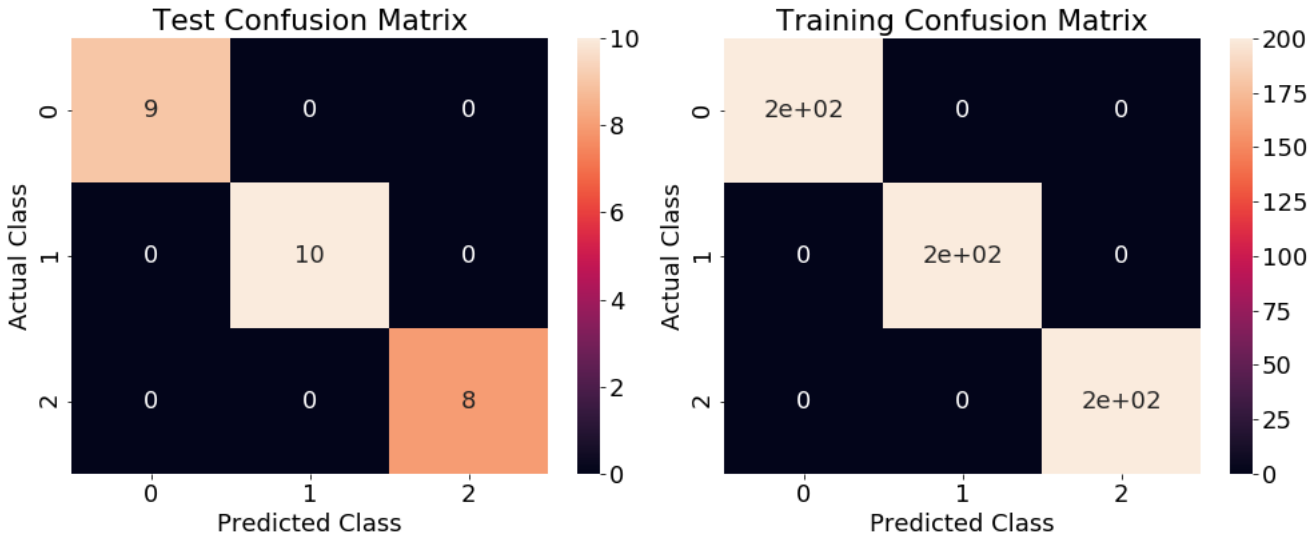


Figure 12: Confusion matrices for data 1B - diagonal covariance matrices

The decision region plot and the contour plot for the best model is figure 13

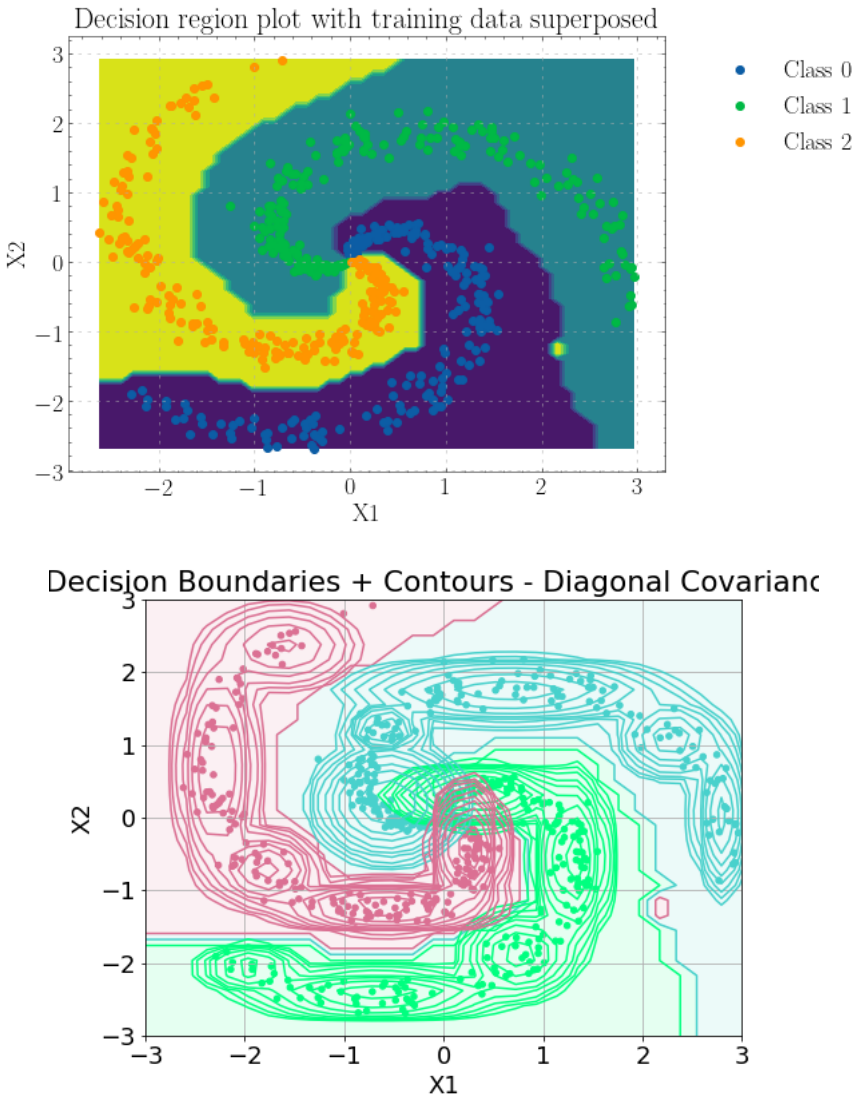


Figure 13: Decision region plot for Bayesian GMM model using diagonal covariance matrix and 5 gaussian components on dataset 1B

2.4 Bayes Classifier with knn

While the main principle remains the same as discussed in section **??**, we now use non-parametric methods to evaluate the class conditional probability $p(\vec{x}/y_i)$.
Suppose the number of data points in the hyper-volume v around $\vec{x}$ be N, the number of data point corresponding to class i be N_i, then:

$$p(\vec{x}/y_i) = \frac{N_i}{N * V} \tag{8}$$

The probability density can be estimated in two ways :

- Specifying the volume V, number of point N_i and N are calculated
- Specifying N, the volume V and N_i are calculated. Here, the radius of hyper sphere is the distance of the point belonging to N farthest from $\vec{x}$, this is called knn method.

For this case, we use the knn method to evaluate the class conditional probability densities.
The class label i that maximizes equation 2 is chosen as the label for $\vec{x}$.

2.4.1 Model Performance for varying values of k

The model is tested for k=10 and k=20. The accuracy table and confusion matrix are as follows:

k-value	Train accuracy	validation accuracy	Test Accuracy
10	0.991667	1.0	0.955556
20	0.986667	1.0	0.933333

Table 4: Accuracy table for data set 2a- Bayes Classifier with knn

Since the accuracy is higher for k=10, it is used to further evaluate the confusion matrix and decision boundary plot.

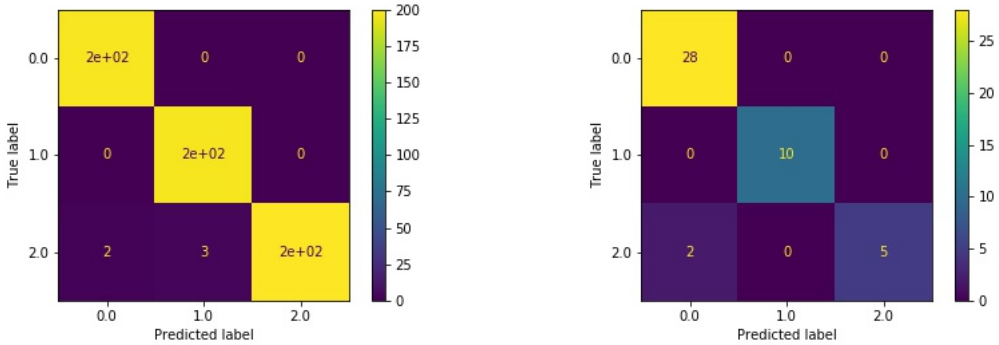


Figure 14: Confusion matrices for k=10, for train data and test data on left and right respectively.

2.4.2 Decision boundary plot

Figure 15: Decision boundary for k=10

While the shape of decision boundary is almost similar as figure 4, there are still some differences, especially near the edges.

3 Dataset 2A

3.1 Bayes Classifier, GMM, full covariance

3.1.1 Training and Validation Accuracy

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

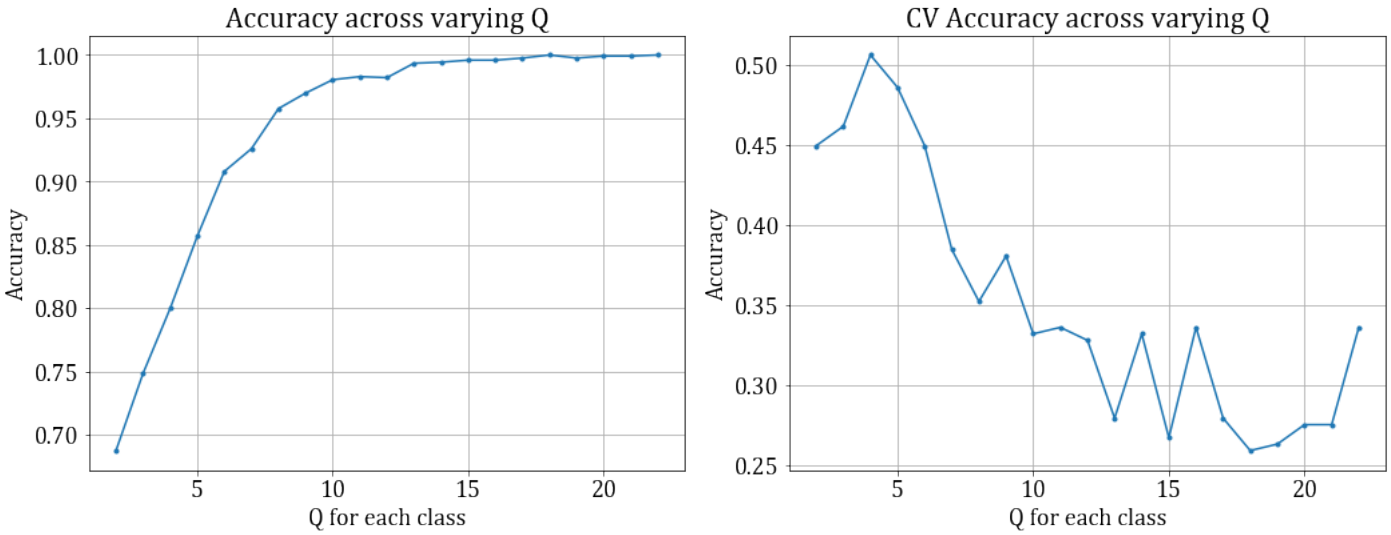


Figure 16: Training and Validation accuracy across q_i, on the left and right respectively

3.1.2 Testing Accuracy

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

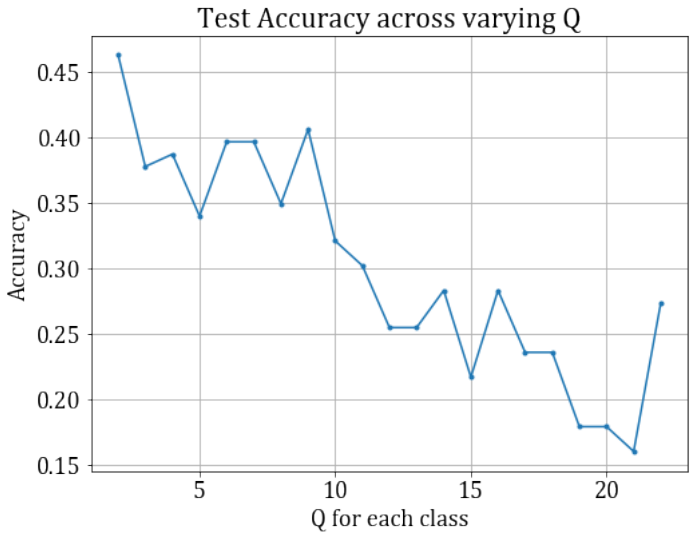


Figure 17: Testing accuracy across q_i

3.1.3 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 6. The accuracies obtained in tabular format is as follows:

# Clusters/Class (Q)	Train Accuracy	Validation Accuracy	Test Accuracy	Sum(Train,Validation)
2	0.687805	0.449393	0.462264	1.137198
3	0.748780	0.461538	0.377358	1.210319
4	0.800000	0.506073	0.386792	1.306073
5	0.856911	0.485830	0.339623	1.342741
6	0.908130	0.449393	0.396226	1.357523
7	0.926016	0.384615	0.396226	1.310632
8	0.957724	0.352227	0.349057	1.309950
9	0.969919	0.380567	0.405660	1.350486
10	0.980488	0.331984	0.320755	1.312472
11	0.982927	0.336032	0.301887	1.318959
12	0.982114	0.327935	0.254717	1.310049
13	0.993496	0.279352	0.254717	1.272848
14	0.994309	0.331984	0.283019	1.326293
15	0.995935	0.267206	0.216981	1.263141
16	0.995935	0.336032	0.283019	1.331967
17	0.997561	0.279352	0.235849	1.276913
18	1.000000	0.259109	0.235849	1.259109
19	0.997561	0.263158	0.179245	1.260719
20	0.999187	0.275304	0.179245	1.274491
21	0.999187	0.275304	0.160377	1.274491
22	1.000000	0.336032	0.273585	1.336032

Table 6: Variation of accuracy across hyperparameter values on the training, validation and test set using the GMM model with full covariance matrix on Dataset 2A. The row corresponding to the best model has been highlighted.

The confusion matrix obtained for the model with $q_i = 6$ are as follows:

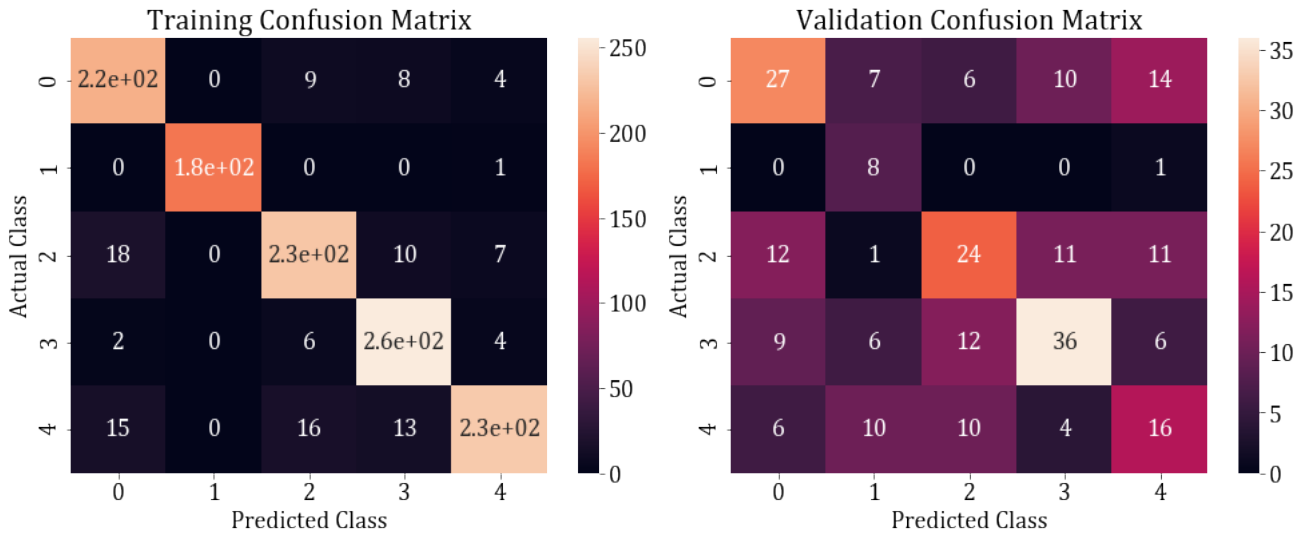


Figure 18: Training and Validation confusion matrices for the best model with $q_i = 6$, on the left and right respectively

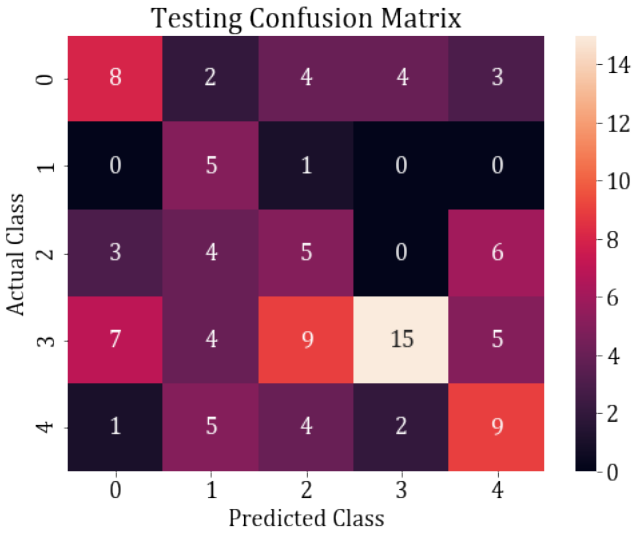


Figure 19: Testing confusion matrix for the best model with $q_i = 6$.

In addition to just taking the same number of clusters for all classes, a parameter sweep was done to identify the best combination of cluster numbers for the dataset. The accuracies obtained from the parameter sweeps are as follows:

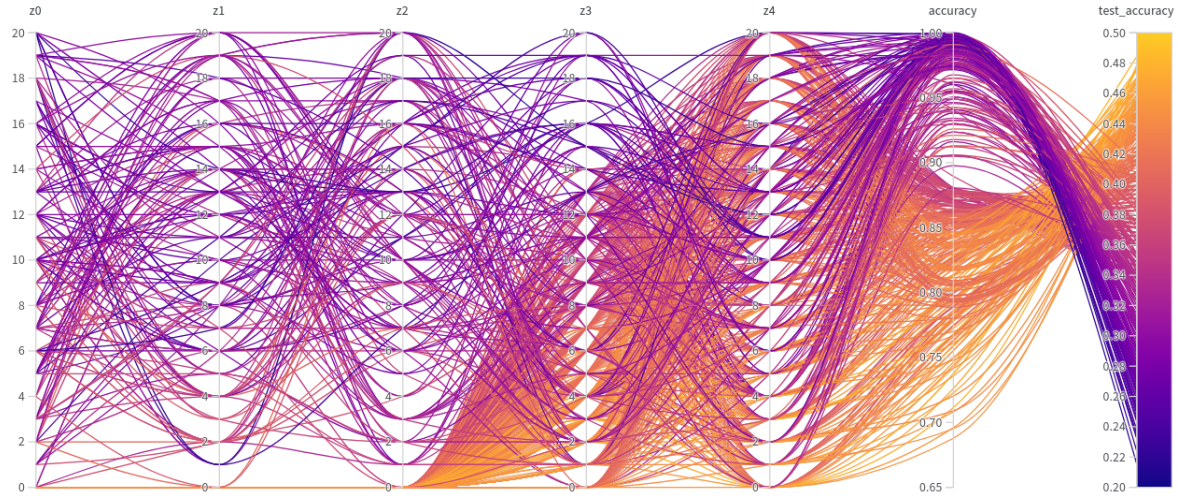


Figure 20: Parameter Sweep Results for the dataset 2A.

From the graph above, the parameter combination that resulted in the best validation accuracy is:

- $q_1 : 2$
- $q_2 : 2$
- $q_3 : 2$
- $q_4 : 6$
- $q_5 : 3$

The accuracies obtained are as follows:

- Training accuracy: 0.7853658536585366
- Validation accuracy: 0.4939271255060729
- Testing accuracy: 0.46226415094339623

From the results above, we can see that the validation and test accuracies are higher in this case, however, the train accuracy is low.

The confusion matrices obtained are as follows:

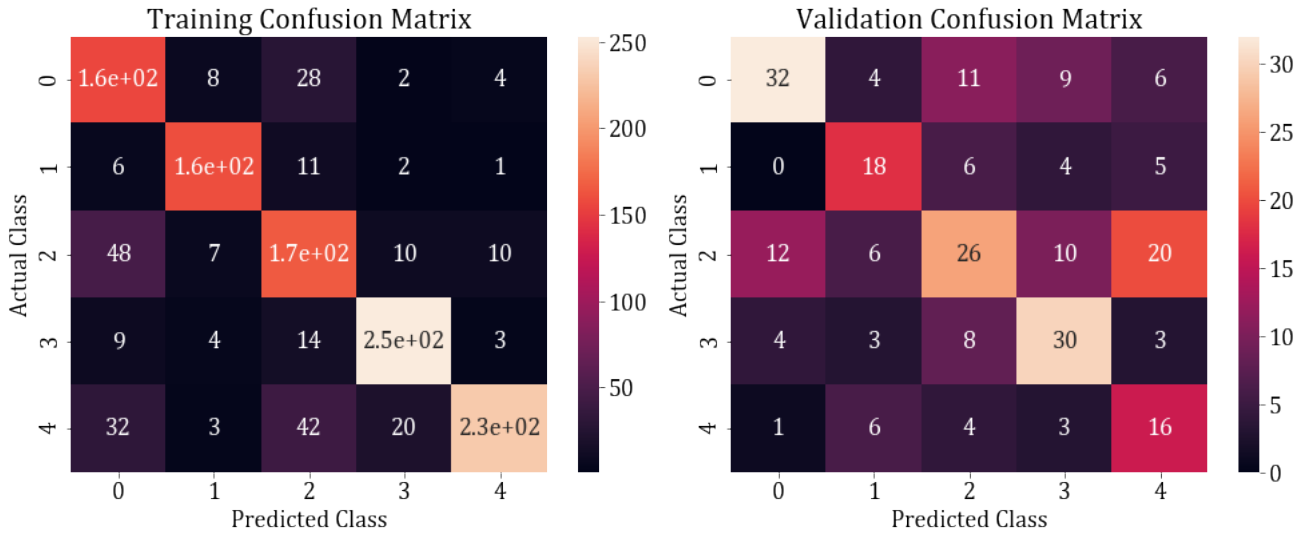


Figure 21: Training and Validation confusion matrices for the model with varying q_i, on the left and right respectively

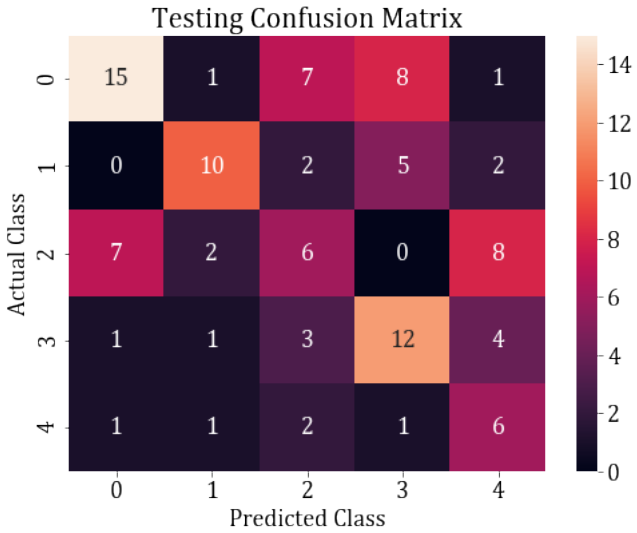


Figure 22: Testing confusion matrix for the model with varying q_i.

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 7. The plot of the same is in figure 23. The tolerance 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 7: Variation of accuracy across hyperparameter values on the training and validation using the GMM model with diagonal covariance matrix on Dataset 2A.

Figure 23: Accuracy for training and validation set for 2A

3.2.2 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 is figure 24 .

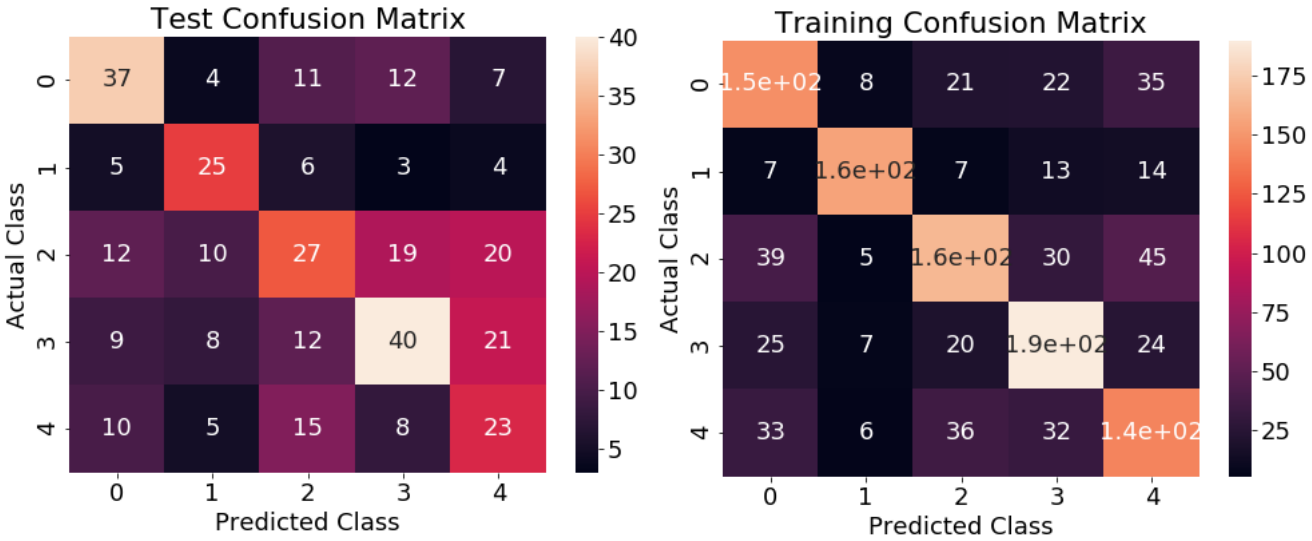


Figure 24: Confusion matrices for data 2A - diagonal covariance matrices

4 Dataset 2B

The varying length classification is done as follows:

- Each image has a $36 * 23$ feature parameters.
- Each of these 36 rows are considered as instances of the class and a GMM (full covariance/diagonal covariance) is fit on the dataset.
- The E-step and M-step are done as is in case of static length pattern.
- However, for each of these images,

$$p(X|\lambda_i) = \prod_{t=1}^{T} p(x_t|\lambda_i) = \prod_{t=1}^{T} \sum_{q=1}^{Q} w_{iq} \mathcal{N}(x_t|\mu_{iq}, \Sigma_{iq}) \tag{9}$$

  is calculated to perform classification.

Due to the size of the dataset, we weren't able to completely train the models of the five classes.