PRNN ASSIGNMENT 1

Kalyan Reddy
SR. No. 21361

Ramesh Babu
SR. No. 20950

Abstract

This report consists the results and interpretations of a Assignment 1. Python code for this assignment is in ***PRNN_Assignment_1_Ramesh_Kalyan.ipynb***

1. Problem 1: Linear Regression

DATASET : ***p1train.csv, p1test.csv***

Linear regression to predict the current health status of an organism based on the measurements obtained from two biological sensors that measure their bio-markers. The target variable indicates the health status, and negative values denote below-average health conditions. The aim is to build a linear regression model using the training data and evaluate its performance on the test split using the following metrics:
(a) Mean Squared Error(MSE)
(b) Mean Absolute Error (MAE)
(c) p-value out of significance test

1.1. Algorithm

Parameter Update using Gradient Descent using Exact Line Search where f is Empirical Risk function

$$w_{i+1} = w_i - \alpha \nabla f(w_i) \tag{1}$$

Learning Parameter Update Equation is given by

$$\alpha_k = \frac{||\mathbf{g}_k||^2}{\mathbf{g}_k^\top \mathbf{Q} \mathbf{g}_k} \; where \; \mathbf{g}_k = \nabla \mathbf{f}(\mathbf{w_i})$$

which is derived from

$$\alpha^* = \arg\min_{\alpha \geq 0} f(w_k + \alpha \Delta w_k) \tag{2}$$

$$\mathbf{Q} = \mathbf{X}^\top \mathbf{X} \tag{3}$$

1.2. Results

The following are the metrics computed:

Metric	Value
MSE	5.046436003951332
MAE	1.799080053098278
p-value	0.9084780712922236

Table 1. Results on Test data

Metric	Value
MSE	5.059684615643717
MAE	1.791753209393997
p-value	0.999999999998459

Table 2. Results on Train data

1.3. Plots

Plot of Linear Regression Plane

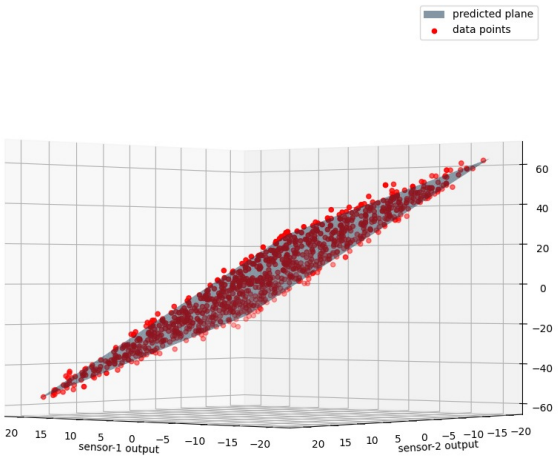

Figure 1. Regression Plane with sample data points

1.4. Inferences

A p-value of 0.9999 for the train data suggests that the null hypothesis is highly likely to be true. This means that the observed result is not significant and is likely to occur by chance. In other words, there is no evidence to reject the null hypothesis.

A p-value of 0.9084 for the test data suggests that the null hypothesis is less likely to be true. This means that the observed result is statistically significant and is less likely to occur by chance. In other words, there is evidence to reject the null hypothesis.

Mean Square Error and Mean Absolute Error are almost same for both Train and Test data

2. Problem 2: Non Linear Models

A kernel function, denoted by ϕ, is used to map the input features(n dim) into a higher dimensional space(m dim), where a linear model can be used to fit the data

$$\phi : \mathbb{R}^n \to \mathbb{R}^m \tag{4}$$

The algorithm is same as the first algorithm(p1), the data is non-linearlized using three kernels namely $\phi_{\mathbf{1}}, \phi_{\mathbf{2}}, \phi_{\mathbf{3}}$

2.1. Implementation

The dataset is min-max normlized , which scales the features between 0 and 1.

$$x_{normalized} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

The normalized three non-linear functions.

$$\phi_{\mathbf{1}} : \mathbb{R}^4 \to \mathbb{R}^7$$

$$\phi_1 = \begin{pmatrix} x_0 & x_1^2 & x_2^2 & x_3^2 & x_1x_2 & x_2x_3 & x_1x_3 \end{pmatrix}^\intercal$$

$$\phi_{\mathbf{2}} : \mathbb{R}^4 \to \mathbb{R}^4$$

$$\phi_2 = \begin{pmatrix} x_0 & \sin^{-1}(x_1) & \cos^{-1}(x_2) & \tan^{-1}(x_3) \end{pmatrix}^\intercal$$

$$\phi_{\mathbf{3}} : \mathbb{R}^4 \to \mathbb{R}^6$$

$$\phi_3 = \begin{pmatrix} x_0 & e^{x_1} & \tan^{-1}(x_2) & x_2x_3 & e^{x_1^2} & e^{x_2}\sin^{-1}(x_1) \end{pmatrix}^\intercal$$

where $\boldsymbol{x}$ is the input data with dimensionality 4 and 7, 4, 6 are the dimensions of the outputs of kernels' data respectively.

2.2. Results

The following are the Metrics for 3 non linear kernels

Metric	Value
MSE	0.003809012158144592
MAE	0.04466949505623271
p-value	0.6258766958156505

Table 3. Results on Test data for polynomial kernel

Metric	Value
MSE	0.013106455044916317
MAE	0.07841319517681597
p-value	0.0460311640617559

Table 4. Results on Test data for sinusoidal kernel

Metric	Value
MSE	0.006148542926184597
MAE	0.051525256141034974
p-value	0.0020269734427482036

Table 5. Results on Test data for exponential kernel

2.3. Interpretations

Inferring from the above results, it is evident that kernel 1 performed better as MSE is low for kernel 1 compared to others. Sinusoidal kernel performed worse. hence the data might be from a polynomial function however combinations of exponential and polynomial performed decent among the three.

3. Problem 3 : Predicting Products from Sensor Data

The objective here is to classify the products being produced by an industry into 5 classes based on the data from the 10 sensors fitted. The implementation involves the use of Bayes' classifiers with 0-1 loss assuming Normal, Laplacian, and GMMs (with diagonal co-variances) as class-conditional densities. Additionally, a linear classifier is implemented using the one-vs-rest approach and a multi-class logistic regressor with gradient descent.

3.1. Implementation

3.1.1 MLE Parameters for Normal Density:

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{5}$$

$$\hat{\mathbf{\Sigma}} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^T \tag{6}$$

3.1.2 MLE Parameters for Laplacian Density:

$$\hat{\mu} = \text{median}(x_1, x_2, ..., x_n) \tag{7}$$

$$\hat{b} = \frac{1}{n} \sum_{i=1}^{n} |x_i - \hat{\mu}| \tag{8}$$

3.1.3 GMM Update Steps:

E step: Evaluate the responsibilities using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j)} \tag{9}$$

M step: Re-estimate the parameters using the current responsibilities:

$$\mu_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n \tag{10}$$

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k^{new})(x_n - \mu_k^{new})^T \tag{11}$$

$$\pi_k^{new} = \frac{N_k}{N} \tag{12}$$

where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \tag{13}$$

Log-Likelihood:

$$\ln p(X|\mu, \Sigma, \pi) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \right) \tag{14}$$

3.1.4 Linear one vs rest and Logistic Regression

Both "Logistic One vs Rest" and Softmax(Multiclass Logistic Regression) are popular approaches for multiclass classification using logistic regression.

Logistic One vs Rest (OvR): In this approach, a binary logistic regression model is trained for each class against the rest of the classes. For instance, if there are three classes (A, B, and C), we will train three logistic regression models: one for class A versus classes B and C, one for class B versus classes A and C, and one for class C versus classes A and B. During prediction, the model with the highest probability is chosen as the predicted class. Softmax regression, on the other hand, directly models the probability distribution over all classes using a single classifier. The model outputs a vector of probabilities for each input, with each element of the vector representing the probability of the input belonging to a particular class.

Both approaches have their advantages and disadvantages:

OvR is simpler to implement and scales well to a large number of classes. It also works well when the classes are well separated. However, it may struggle when the classes are not balanced, and one class has a much larger number of samples than others. OvR logistic regression can be effective when the classes are imbalanced, as it creates a separate classifier for each class. This can help to ensure that each class is well-represented in the model.

In summary, the choice between OvR and softmax regression depends on the specific problem at hand, such as the number of classes, class balance, and the level of overlap between the classes.

3.2. Results

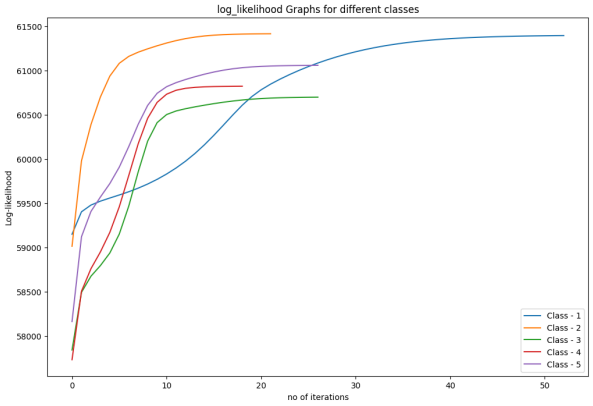

Figure 2. LogLikelihood curves for different classes with 3 components

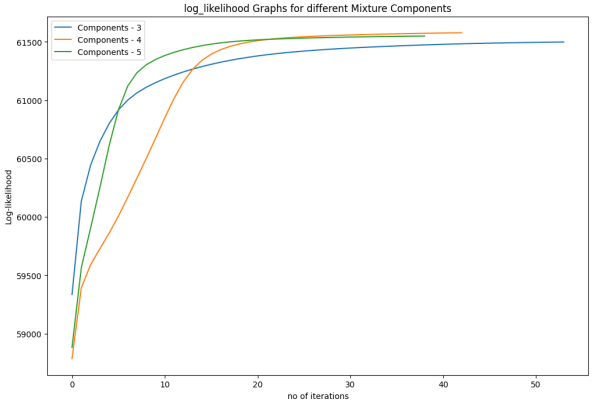

Figure 3. LogLikelihood curves for a class using different mixtures

Parameter	Normal Density	Laplacian Density	GMM(n = 3)	GMM(n = 8)
Classification Accuracy	59.3866%	56.706 %	58.879 %	58.68 %
Confusion Matrix	1629 363 261 345 373 381 1746 277 269 308 299 298 1912 240 291 321 335 253 1796 270 364 310 231 303 1825	1481 415 285 360 430 377 1700 291 274 339 320 316 1838 264 302 338 350 261 1741 285 388 323 245 331 1746	1649 385 244 332 361 406 1747 245 276 307 347 296 1854 231 312 366 329 241 1762 277 390 325 219 279 1820	1655 373 241 334 368 419 1732 242 269 319 368 289 1852 224 307 383 319 232 1748 293 393 324 207 294 1815
F1 scores	0.54618609 0.57881651 0.64010713 0.60593792 0.59836065	0.50417021 0.55875102 0.61677852 0.58570227 0.56919315	0.53809756 0.57628236 0.63460551 0.60187873 0.59574468	0.53481984 0.57560651 0.63708290 0.59822039 0.59168704

Table 6. Metrics for Normal, Laplacian, GMMs

Parameter	Linear classifier using the one-vs-rest approach	Multi-Class Logistic Regression
Classification Accuracy	57.32%	57.326%
Confusion Matrix	1698 312 285 323 363 304 1897 259 287 293 343 269 1749 288 326 326 265 344 1759 339 391 311 383 391 1495	1518 390 297 375 391 377 1696 297 286 325 308 306 1881 256 289 339 345 262 1739 290 350 327 253 338 1765
F1 scores	0.5619725 0.6225795 0.5834862 0.5785232 0.5166753	0.51782363 0.56112489 0.62388059 0.58267716 0.57935335
Empirical Risk on Train data	2.961474	1.143054
Empirical Risk on Test data	2.977265	1.140575

Table 7. Metrics for Logistic Regression

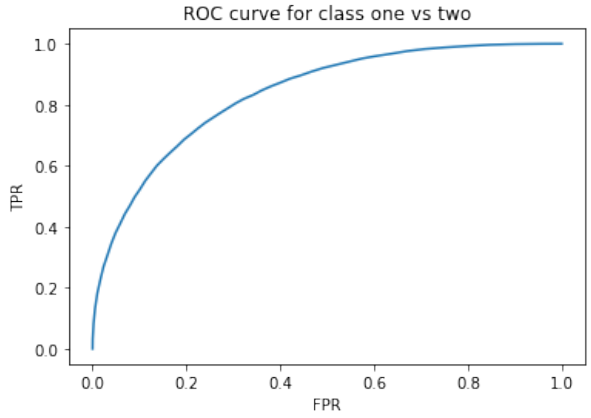

Figure 4. ROC Curve for class 1 and 2 using MultiClass Logistic Regression

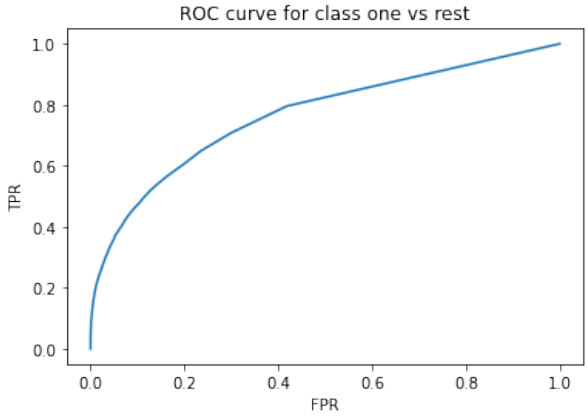

Figure 5. ROC Curve for class 1 vs Rest using linear 1 vs Rest approach

3.3. Inferences

- Bayes' Classifier with class conditional densities modelled as Normal, Laplacian, GMMs gives an accuracy of **59.38%**, **56.76%** and **58.88%** respectively. Performance Metrics are slightly higher in case of Normal, GMMs that that of Laplacian Model. This indicates that the data given maybe sampled from Normal or GMMs. Also when the mixture components are changed, no significant change is observed in performance metrics and are similar to Normal Density metrics. This indicates that the sampling is done from a single Normal density

- Computation of GMMs becomes heavier when the mixture components are increased which is obvious as the no. of parameters increase. Loglikelihood takes more iterations to converge for class 1 when compared to other classes with various mixture components which may indicate that the class conditional density of class 1 is more complex than that of other classes since the data is balanced wrt classes

- **Comparision between Linear One Vs rest vs Logistic classifier**:

  The accuracies for both appraches are almost same, this might be due to following reasons:

- **Similar model assumptions**: Both OvR logistic regression and softmax regression assume that the output

probabilities are a linear function of the input features. This assumption can be appropriate for many classification problems and may contribute to the similar accuracies.

- **Similar training objectives**: Both OvR logistic regression and softmax regression aim to minimize the negative log-likelihood of the observed data. While the optimization algorithms may be different, the similarity in the objectives can lead to similar model performance.
- **Similar decision boundaries**: OvR logistic regression and softmax regression can produce similar decision boundaries between classes, depending on the nature of the classification problem and the features used. This can result in similar prediction accuracies for the two methods. However, it's important to note that there can be situations where OvR logistic regression and softmax regression can give different accuracies. For example, softmax regression may be more suitable for problems where the classes are not well-separated or there is significant overlap between them. Ultimately, the choice of method should be based on the specific characteristics of the classification problem at hand.

4. Problem 4: Kannada MNIST

DATASET : ***p4_train.csv(from kaggle)***

The objective of this problem is to classify the images of Kannada MNIST dataset into 10 classes. Here, we have used the image data in the form of csv files from Kaggle. The followind classification schemes are tested with this data: Naive Bayes' with Normal as Class conditional, Logistic regressor with gradient descent, and Multi-class Bayes' classifier with GMMs with diagonal co-variances for class conditionals. And the same experiment is repeated with various split ratios of train and test components of data

4.1. Implementation

4.1.1 Naive Bayes' with Normal as Class Conditional

The feature vectors are treated as independent to each other in this case and the individual means and standard deviations are calculated for every features and the class conditional densities are found. Bayes' classifier is implemented using these class conditionals and priors
Posterior probabilities are calculated using Bayes' rule:

$$P(Y \mid X_1, X_2, ..., X_n) = \frac{P(Y) \cdot \prod_{i=1}^{n} P(X_i \mid Y)}{\prod_{i=1}^{n} P(X_i)} \quad (15)$$

Where:

- Y represents the class label.
- $X_1, X_2, ..., X_n$ are the predictor variables.
- $P(Y)$ is the prior probability of class Y.
- $P(X_i \mid Y)$ is the conditional probability of predictor variable X_i given class Y.
- $P(X_1, X_2, ..., X_n)$ is the probability of the predictor variables.

4.1.2 Multi-class Logistic Regression:

The data has been split into the given ratios using random split.
The data is normalized using mini-max normalization as it performed better than mean-std normalization. This is because when the data has a known range, Min-max normalization scales the data to a known range, typically between 0 and 1, which can be useful when the data has a known minimum and maximum value. For example, when working with image data, the pixel values are typically between 0 and 255, and min-max normalization can be used to scale the values to the range [0,1]. In this case, standardization may not be appropriate as it could distort the information contained in the data.

Overall, the choice between min-max normalization and standardization depends on the specific characteristics of the data and the problem being addressed. It is often a good idea to try both methods and compare their performance empirically.
Soft max function used for Multi-class Logistic Regression is:

$$P(Y = K \mid X = x) = \frac{e^{\omega_{0k} + \omega_{1k} x_1 + ... + \omega_{pk} x_p}}{\sum_{j=1}^{m} e^{\omega_{0j} + \omega_{1j} x_1 + ... + \omega_{pj} x_p}} \quad (16)$$

Where:

- K represents the class labels 1, 2, 3 ....m
- m is the number of classes.
- $P(Y = K \mid X = x)$ is the probability of the observation x belonging to class K.
- $\omega_{0k}, \omega_{1k}, ..., \omega_{pk}$ are the coefficients for class K.
- $x_1, ..., x_p$ are the predictor variables.

4.1.3 GMMs as Class Conditional

GMMs are implemented using the update equations (9) to (14). The data is normalized using min-max normalization. The same experiment is repeated for various data split ratios.

Metric	*20:80*	*30:70*	*50:50*	*70:30*	*90:10*
Classification Accuracy	84.09375 %	82.36666 %	77.59333 %	81.39444 %	78.7 %
F1 scores	$\begin{bmatrix}0.8829\\0.9277\\0.9480\\0.6538\\0.8606\\0.8522\\0.7211\\0.6870\\0.9290\\0.9422\end{bmatrix}$	$\begin{bmatrix}0.9048\\0.9586\\0.9655\\0.5381\\0.8214\\0.8979\\0.7053\\0.6411\\0.9345\\0.8386\end{bmatrix}$	$\begin{bmatrix}0.8117\\0.7747\\0.9691\\0.6279\\0.7723\\0.8413\\0.5568\\0.5810\\0.9382\\0.9121\end{bmatrix}$	$\begin{bmatrix}0.7382\\0.6289\\0.9726\\0.7854\\0.8951\\0.8747\\0.6817\\0.6623\\0.9229\\0.9502\end{bmatrix}$	$\begin{bmatrix}0.7213\\0.5143\\0.9744\\0.7709\\0.8885\\0.8615\\0.5099\\0.6398\\0.9402\\0.9521\end{bmatrix}$

Table 8. Metrics of Naive Bayes' Classifier with Normal Class Conditionals

Metric	*20:80*	*30:70*	*50:50*	*70:30*	*90:10*
Classification Accuracy	83.75625 %	95.123 %	95.086 %	95.077 %	95.166 %
F1 scores	$\begin{bmatrix}0.9465\\0.9664\\0.9858\\0.9252\\0.9496\\0.9570\\0.9296\\0.8974\\0.9758\\0.9601\end{bmatrix}$	$\begin{bmatrix}0.9460\\0.9672\\0.9855\\0.9282\\0.9530\\0.9582\\0.9326\\0.9019\\0.9765\\0.9628\end{bmatrix}$	$\begin{bmatrix}0.9468\\0.9672\\0.9864\\0.9306\\0.9511\\0.9585\\0.9315\\0.9033\\0.9730\\0.9606\end{bmatrix}$	$\begin{bmatrix}0.9459\\0.9681\\0.9859\\0.9274\\0.9532\\0.9631\\0.9261\\0.8990\\0.9742\\0.9653\end{bmatrix}$	$\begin{bmatrix}0.9388\\0.9691\\0.9895\\0.9305\\0.9529\\0.9590\\0.9395\\0.9027\\0.9701\\0.9640\end{bmatrix}$
Empirical risk on Train data	0.15736	0.17329	0.1762	0.1786	0.1803
Empirical risk on Test data	0.1950	0.1881	0.1885	0.1910	0.1876

Table 9. Metrics of Multi-class Regression

4.2. Results

4.2.1 Logistic Regression Accuracies and Empirical Risk

From the table, we can observe that accuracies and risks for train and test data is almost same for every split ratio (infer Table 8)

It is not guaranteed that the empirical risk will be the same for all ratio splits of training and test data in multi-class logistic classification. The empirical risk depends on various factors such as the distribution of the data, the choice of hyperparameters, and the choice of Loss function.

However, when the data is large and the distribution of the data is relatively stable, we can expect that the empirical risk will be similar across different random splits of the data into training and test sets. This is because with larger amounts of data, the models will have more opportunities to learn from the data and will be less affected by small variations in the specific data points used for training and testing.

In practice, it is still a good idea to use techniques such as cross-validation to estimate the empirical risk more robustly, even when the data is large. Cross-validation involves splitting the data into multiple subsets and using each subset in turn as a validation set, while using the remaining subsets as the training set. By repeating this process multiple times with different folds, we can obtain a more reliable estimate of the empirical risk that is less sensitive to the specific train-test split.However due to time bounds for the deadline we could not do it.

4.2.2 ROC curves

One vs. Rest (OvR) and One vs. One (OvO) are two methods we used for ROC analysis to multi-class problems.

The number of classes in a multi-class classification problem can affect the performance of both the One vs. Rest (OvR) and One vs. One (OvO) approaches.

In the One vs. Rest approach, For example, if there are 5 classes, 5 the classfier considers as(class 1 vs rest, class 2 vs rest, class 3 vs rest, class 4 vs rest, and class 5 vs rest). The ROC curve for one classifier is plotted(for class 5), as shown in the figure

We have got similar ROC curves in both the cases. The

plot is for class 5 vs rest but similar results were obtained for all the classes.The AUC was very good in both the cases as one can see from the plots. This can be confirmed by the accuracies as well, as the accuracies were around 95% which indicates the classifier is good.

In general, OvR may be more prone to class imbalance issues than OvO, as it can lead to imbalanced datasets in the binary classification problems that are created. This can potentially affect the ROC curve performance. On the other hand, OvO can be more computationally expensive, as it requires training more binary classifiers.

However, there are situations where the OvR and OvO ROC curves may have similar performance. For example, if the classes are relatively balanced which is in our case and the binary classifiers are able to accurately distinguish between the classes, both approaches may lead to similar ROC curves.

In summary, while it's possible that the OvR and OvO ROC curves may have similar performance, this is not always the case and depends on various factors related to the specific problem and dataset.

4.2.3 Plots

The following are the ROC curves for two apporaches

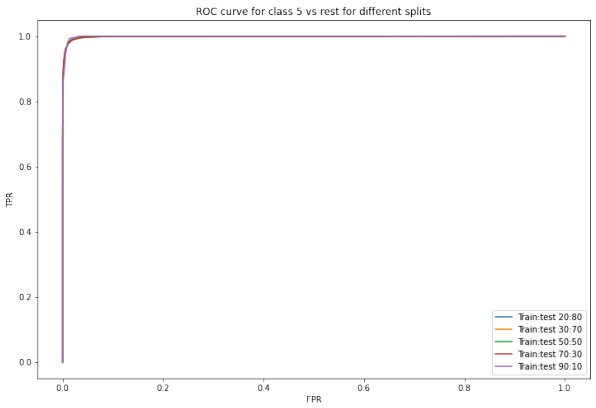

Figure 6. ROC curve for class 5 vs rest

4.3. Inferences

Naive Bayes' with Normal as Class Conditionals

We can observe that the accuracies are fluctuating in no specific manner when the split ratio is changed. This is because the data given has 784 features and while calculating the posterior probabilities, we are multiplying all the 784 values returned by the univariate Normal densities at that test data point. This makes the posteriors very low. This involves a lot of inaccuracies. Hence it's difficult to calculate the accuracies accurately

Bayes' Classifier with GMMs as Class Conditionals

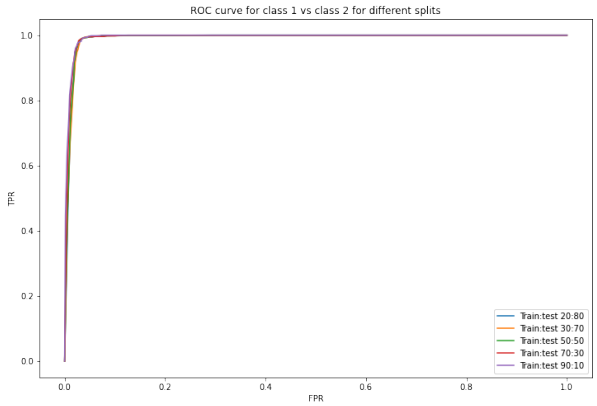

Figure 7. ROC curve for class 0 vs class 1

EM Algorithm cannot be executed on the given 784 dimensional data. While evaluating the responsibilities, the determinant of the Covariance Matrix (at some iteration) becomes zero as we are multiplying 784 very less values returned by 784 dimensional Gaussians. Hence the next updates cannot be found and the EM Algorithm fails. One way to overcome this is by initializing the Covariance Matrices to identities and not update them later. But the convergence and correctness of this Algorithm is to be investigated. However it works for this particular problem

Multi-class Logistic Regression

Overall, Multi class logistic regression performed well on Image data. This can be due to linear decision boundaries, Logistic regression models create linear decision boundaries to separate the different classes. While the MNIST dataset has a large number of features (784), most of these features are highly correlated, and the important information needed to classify the images can be captured with a simple linear model. However, more complex models such as deep neural networks can achieve even better performance on the MNIST dataset and are often used as the state-of-the-art approach.

5. Problem 5: PCA - KANNADA MNIST

DATASET : ***PCA_MNIST.csv***

After applying PCA to the Kannada MNIST dataset, the resulting lower-dimensional representation can be used as input to a machine learning algorithm, such as logistic regression, to perform classification tasks. The reduced dimensionality can lead to improved performance and faster computation times compared to using the original high-dimensional data.

In the case of this MNIST dataset, each image is represented by 784 features (pixels), which can be computationally expensive to process. However, after applying PCA, the dataset can be transformed into a lower-dimensional space

with only the most important information retained. This can significantly reduce the number of features, and thus, the computation time required for training and prediction.

5.1. Multi-class Logistic Regression

The data has been split into the given ratios using random split.
The preprocessing is done similar to the previous problem p4. The computation was quicker than original data because it reduces the dimensionality of the dataset by projecting it onto a lower-dimensional space that captures the most important information in the data. By reducing the number of features in the dataset, the computational complexity of machine learning algorithms is reduced, which leads to faster training and prediction times.

5.2. Results

5.2.1 Logistic Regression Accuracies and Empirical Risk

From the table, we can observe that accuracies and risks for train and test data is almost same for every split ratio. Bt the accuracies are not as good as logistic regression without PCA .This can be due to some following reasons:

- **Loss of Information**: PCA is a dimensionality reduction technique that tries to represent the data with fewer dimensions while retaining as much of the original information as possible. However, by reducing the dimensions, some information might be lost, which could lead to a decrease in the accuracy of the model.
- **Nonlinear Relationships**: Logistic regression assumes a linear relationship between the input features and the output classes. PCA projects the data onto a lower-dimensional space, which might not preserve the nonlinear relationships between the input features and the output classes, leading to a decrease in the accuracy of the model.
- **Overfitting**: PCA can help to reduce the dimensionality of the dataset and remove noise, but it can also create new features that might be spurious. In some cases, these new features might overfit the data, leading to a decrease in the accuracy of the model.
- **Optimal Number of Components**: The number of principal components to be retained after PCA is a hyperparameter that needs to be tuned. If the optimal number of components is not chosen, it could lead to a decrease in the accuracy of the model.

5.2.2 ROC curves

One vs. Rest (OvR) and One vs. One (OvO) are two methods we used for ROC analysis to multi-class problems.

The number of classes in a multi-class classification problem can affect the performance of both the One vs. Rest (OvR) and One vs. One (OvO) approaches.

We have got similar ROC curves in both the cases. The plot is for class 2 vs rest but similar results were obtained for all the classes.The AUC was very good in both the cases as one can see from the plots. This can be confirmed by the accuracies as well, as the accuracies were around 83% which indicates the classifier is good.

In the context of ROC curves, the performance improvement from using PCA on the MNIST dataset with logistic regression may not be very significant. This could result in the ROC curves for logistic regression with and without PCA being relatively similar, with only a small shift towards the upper left corner for the curve with PCA.

This is because the ROC curve represents the trade-off between the true positive rate (TPR) and the false positive rate (FPR) for different classification thresholds, and it's possible that the logistic regression model without PCA is already able to achieve a good balance between TPR and FPR, so reducing the dimensionality further with PCA doesn't provide much additional benefit.

However, it's important to note that the specific shape of the ROC curve can depend on a variety of factors, including the characteristics of the dataset, the choice of model and hyperparameters, and the performance metric being optimized. Therefore, even if the ROC curves for logistic regression with and without PCA are relatively similar, it's still possible that PCA is providing some improvement in performance in other areas, such as reducing overfitting or improving computational efficiency.

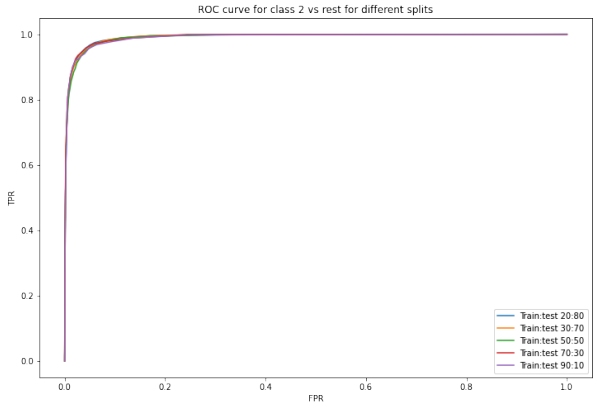

Figure 8. ROC curve for class 5 vs rest

5.3. Inferences

The confusion matrices are in the notebook file.

- **Naive Bayes' with Normal as Class Conditionals** Computationally, Implementation of Naive Bayes'

Metric	*20:80*	*30:70*	*50:50*	*70:30*	*90:10*
Classification Accuracy	85.0458 %	85.2357 %	85.09 %	85.29444 %	84.73333 %
F1 scores	[0.8383 0.8736 0.9383 0.8292 0.8765 0.8987 0.7820 0.6598 0.8888 0.9014]	[0.8406 0.8785 0.9381 0.8290 0.8853 0.8987 0.7818 0.6604 0.8936 0.9024]	[0.8445 0.8804 0.9390 0.8234 0.8785 0.9011 0.7782 0.6601 0.8912 0.8964]	[0.8378 0.8795 0.9452 0.8415 0.8732 0.9076 0.7768 0.6607 0.8878 0.9035]	[0.8049 0.8699 0.9443 0.8473 0.8620 0.8932 0.7808 0.6571 0.8977 0.9004]

Table 10. Metrics of Naive Bayes' Classifier with Normal Class Conditionals

Metric	*20:80*	*30:70*	*50:50*	*70:30*	*90:10*
Classification Accuracy	83.75625 %	82.96428 %	82.3333%	82.4333 %	83.4666 %
F1 scores	[0.8584 0.8937 0.8462 0.8028 0.8859 0.8853 0.7519 0.6478 0.8902 0.8988]	[0.8563 0.8940 0.8254 0.7878 0.8936 0.8931 0.7268 0.6480 0.9057 0.8746]	[0.8643 0.8932 0.8241 0.7698 0.8350 0.8438 0.7634 0.6331 0.8975 0.8878]	[0.8456 0.8757 0.8444 0.7558 0.9007 0.8646 0.7325 0.6475 0.9096 0.8970]	[0.8466 0.8779 0.8588 0.8119 0.8497 0.8653 0.7773 0.6525 0.9226 0.8876]
Empirical risk on Train data	0.642822	0.63030	0.6362	0.64698	0.65367
Empirical risk on Test data	0.66698	0.66830	0.67308	0.65466	0.613140

Table 11. Metrics for Multi-class Logistic Regression

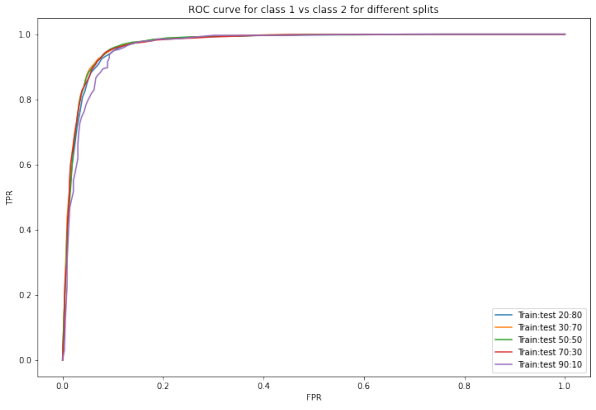

Figure 9. ROC curve for class 0 vs class 1

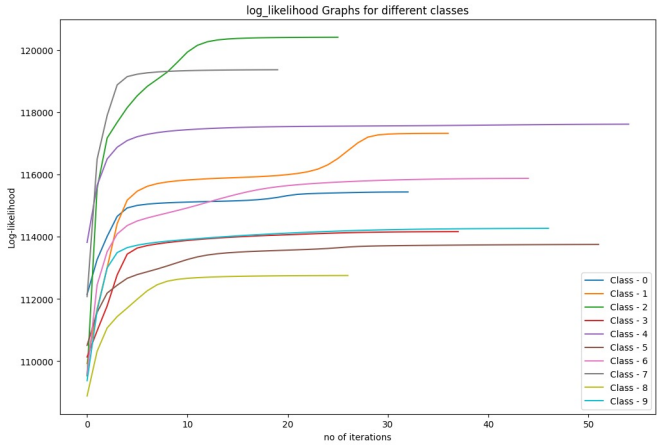

Figure 10. Log-Likelihood curves for 70-30 split, for 5 mixtures for all classes

Classifier on a 10 dimensional data is very light when compared to that on a 784 dimensional data. Also the posterior probabilities calculated are of much high accuracy as we are only multiplying 10 values returned by the 10 dimensional Gaussians

- **Bayes' Classifier with GMMs as Class Conditionals** After reducing the dimensionality to 10 dimensions, EM Algorithm can be executed well and hence we

Metric	*20:80*	*30:70*	*50:50*	*70:30*	*90:10*
Classification Accuracy	68.73 %	70.35 %	68.96 %	64.15 %	68.7
F1 scores	$\begin{bmatrix}0.6180\\0.7330\\0.8260\\0.6004\\0.7934\\0.6326\\0.7229\\0.2530\\0.7952\\0.7125\end{bmatrix}$	$\begin{bmatrix}0.6116\\0.5139\\0.8914\\0.6961\\0.8635\\0.6065\\0.6859\\0.6845\\0.6277\\0.8628\end{bmatrix}$	$\begin{bmatrix}0.6668\\0.7981\\0.6783\\0.4300\\0.8020\\0.5123\\0.7066\\0.4185\\0.6435\\0.6645\end{bmatrix}$	$\begin{bmatrix}0.4989\\0.5550\\0.7606\\0.2947\\0.7155\\0.5045\\0.6981\\0.0356\\0.6032\\0.7697\end{bmatrix}$	$\begin{bmatrix}0.6604\\0.7574\\0.7961\\0.5399\\0.8657\\0.5412\\0.7334\\0.4982\\0.7299\\0.8003\end{bmatrix}$

Table 12. Metrics of Bayes' Classifier with GMMs as Class Conditionals

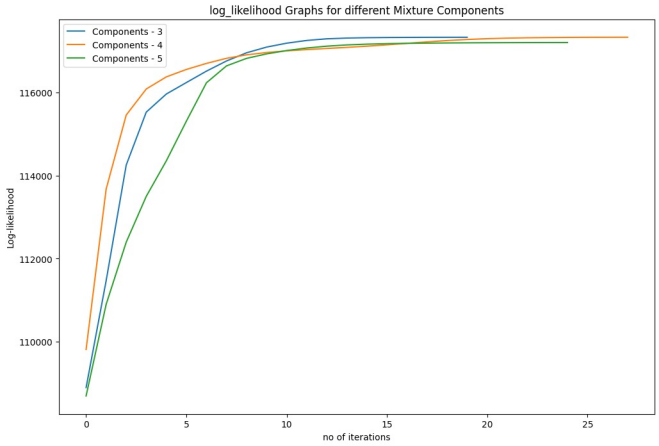

Figure 11. Log-Likelihood curves for different mixtures for class-1

can compute the metrics. The accuracy however is less compared to Naive Bayes' and Logistic Regressor. This could be because of the loss of information due to PCA on 784 dimensional data

- **Multi - class Logistic Regression**

  Overall, After PCA, Multi class logistic regression performed decently on Image data but not as accurate as on original data.