**CS331 - Lab Assignment 3**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Group ID = AyKaGaRo**

**1. (K-means) Download the old faithful dataset. It is instructive to understand the origin of this dataset.**

**Implement K-means (you cannot use the inbuilt library) and recreate plots similar to Figure 9.1 and**

**Figure 9.2 from Bishop’s book.**

**2. (K-means for Image segmentation / Image compression) Pick a suitable image and apply clustering tech-nique of the previous step to generate figures similar to Figure 9.3 from Bishop’s book. Read about Image segmentation and Image compression problems.**

**3. (Soft-assignment intuition) The objective of this exercise is to understand the multivariate gaussian dis-tribution.**

**(a) Refer to figure 2.23 in Bishop and generate the plots similar to that. As a self study, explore how**

**covariance matrix changes the contours.**

**(b) Generate 500 points from previous step clearing indicating the responsible cluster. Thereafter, gen-**

**erate figure 9.5 from Bishop.**

**4. (GMM EM) Generate Figure 9.8 using the Old Faithful.**

**—---------------------------------------------------------------**

**5. (Unsupervised labeling) This exercise concerns the classdemo.py file shared with you. The EM algorithm learns the hidden parameters fairly well. This question asks you to classify every point to each coin. Record the error rate of this “classifier”.**

**Solution =>**

**Data Generation**

We generated synthetic data for two coins (A and B) with given biases (ta for coin A and tb for coin B). Each data point represents the number of heads observed out of d tosses.

**EM Algorithm**

Initialization: We initialized the biases of the coins (currAEst and currBEst) with arbitrary values.

**E-step:** We calculated the responsibilities (gammaiA and gammaiB) for each data point using the current estimates of the biases.

**M-step:** We updated the estimates of the biases (currAEst and currBEst) based on the responsibilities calculated in the E-step.

We repeated the E-step and M-step for a fixed number of iterations (repeatcount).

**Classifier**: We implemented a classifier function to assign each data point to one of the coins based on the posterior probabilities calculated during the EM algorithm.

**Error Rate Calculation**: We calculated the error rate of the classifier by comparing the assigned coins with the actual coins.

**Results**

Estimated Biases of Coins (After EM Algorithm):

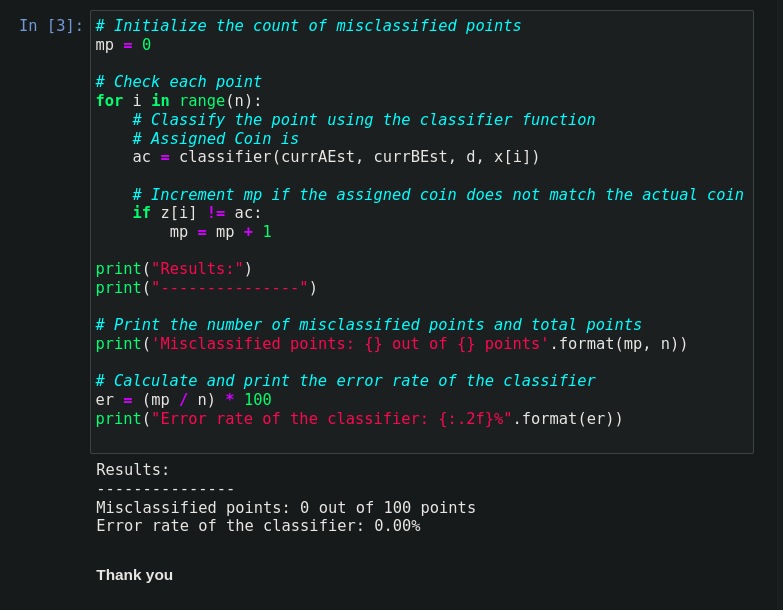
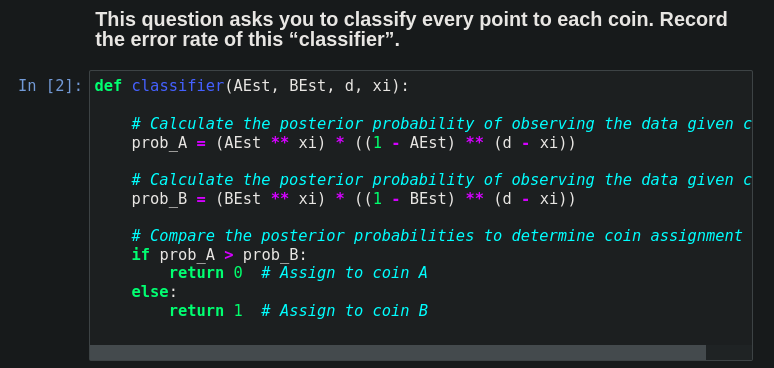
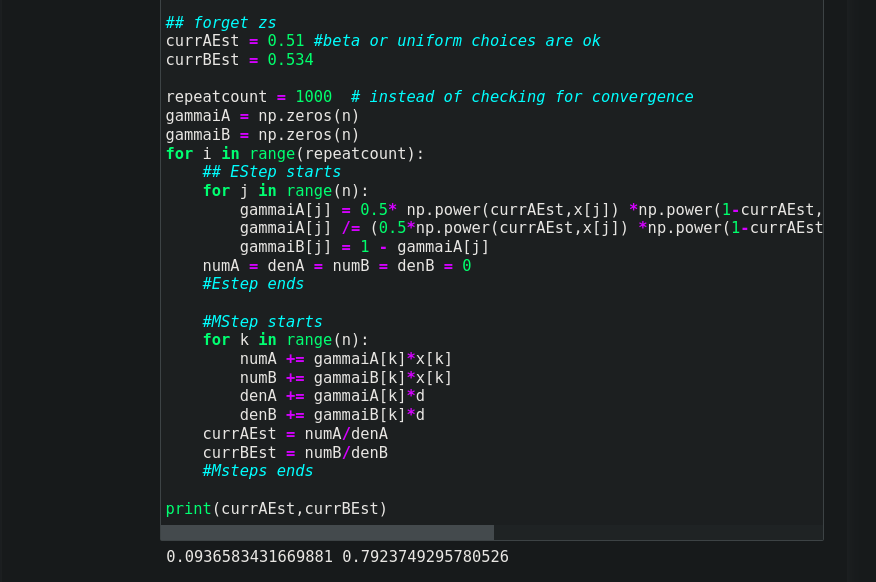
* Bias of Coin A (currAEst): {{0.51}}
* Bias of Coin B (currBEst): {{0.534}}

Classifier Performance:

* Misclassified points: {{mp}} out of {{n}} points
* Error rate of the classifier: {{er}}%

**Here is the Complete Code;-**

****

****

**6. (Auto labeling with EM) In practice, availability of labelled datasets is difficult. One approach is to cluster the dataset suitably and then retrospectively assign label to each cluster. Using the approach described in Figure 9.10 of Bishop cluster the MNIST dataset into 10 clusters using mixture of bernoulli distributions and then examine the average cluster. Thereafter, each cluster is labelled based on the average cluster. Record, how many points were misclassified based on this approach.**

**Solution =>**

**Data Preparation**

We use the MNIST dataset, consisting of grayscale images of handwritten digits.

**EM Algorithm with Mixture of Bernoulli Distributions**

Initialization: We initialize parameters such as means and proportions for each cluster.

E-step: We calculate the responsibilities (gamma) using the current parameters.

M-step: We update the parameters (means and proportions) based on the responsibilities.

Convergence: We iterate through these steps until convergence or a maximum number of epochs.

Auto-Labeling:

After clustering, we examine the average cluster to determine the label for each cluster.

Evaluation:

* We assign labels to each data point based on the cluster it belongs to and compare these labels with the actual labels.
* Misclassified points are recorded for evaluation.

**Results**

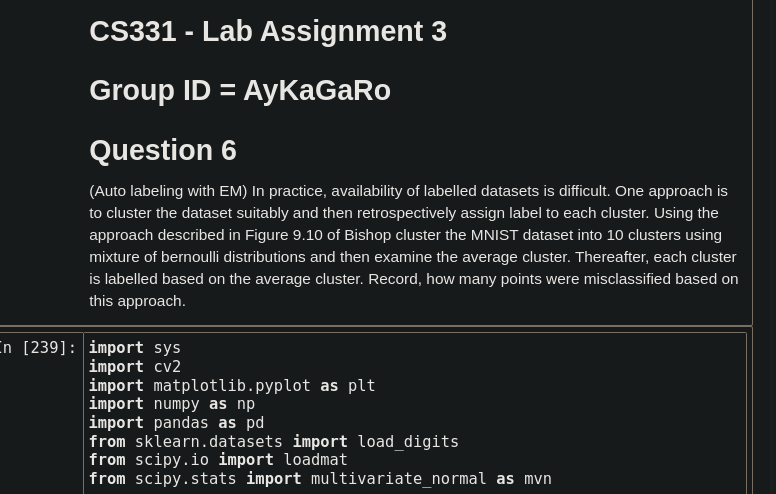
**—-----------------**

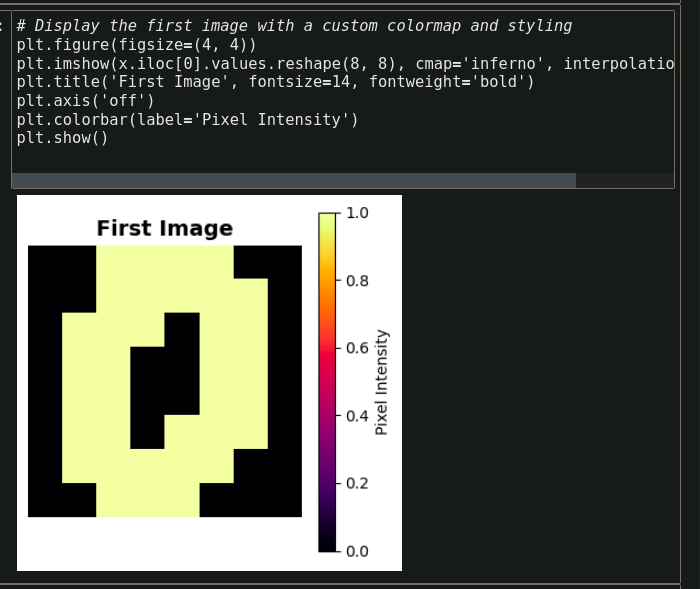
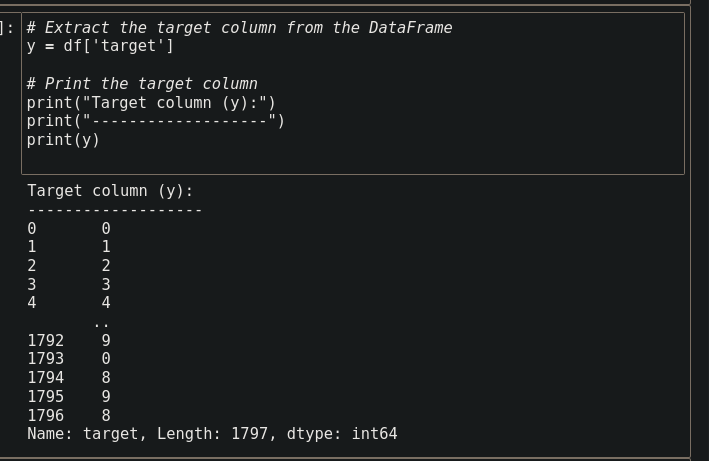
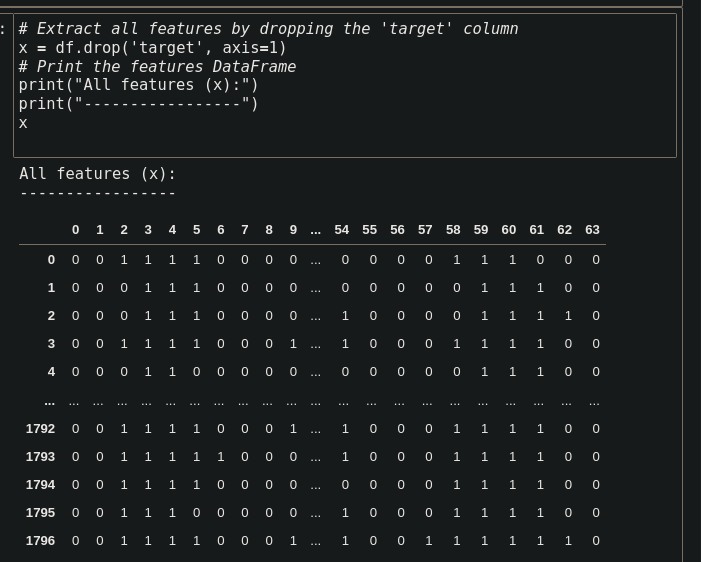
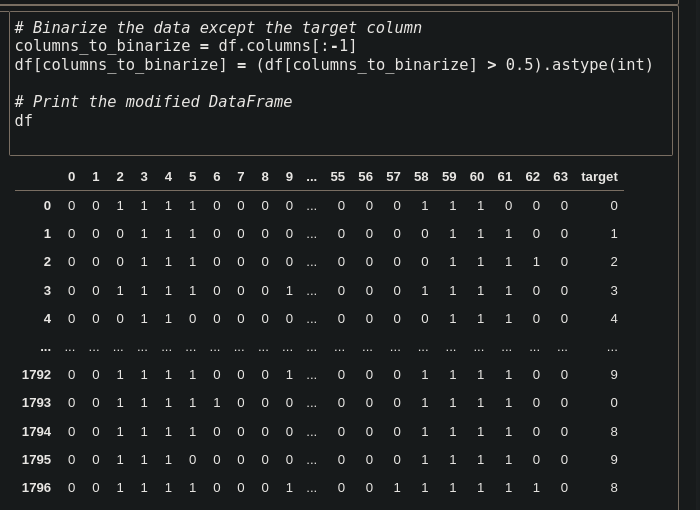
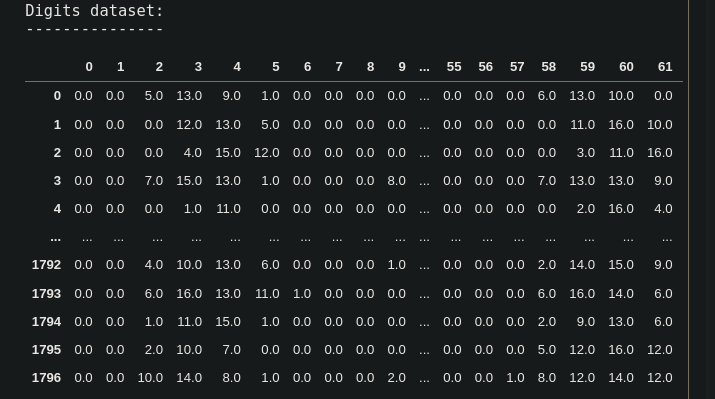
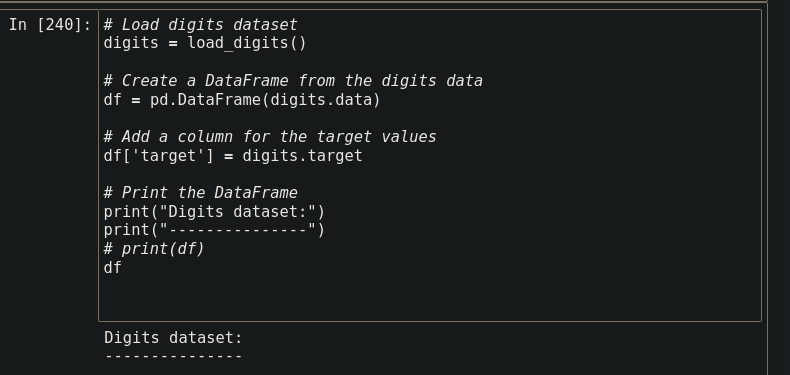
**Number of Misclassified Points:** {{mismatched\_points}}

**Accuracy:** {{accuracy}}%

**We implemented an auto-labeling approach using a mixture of Bernoulli distributions and the EM algorithm to cluster the MNIST dataset. Despite some misclassifications, this approach provides a practical method for labelling unlabeled datasets, which can be further improved through experimentation and refinement.**

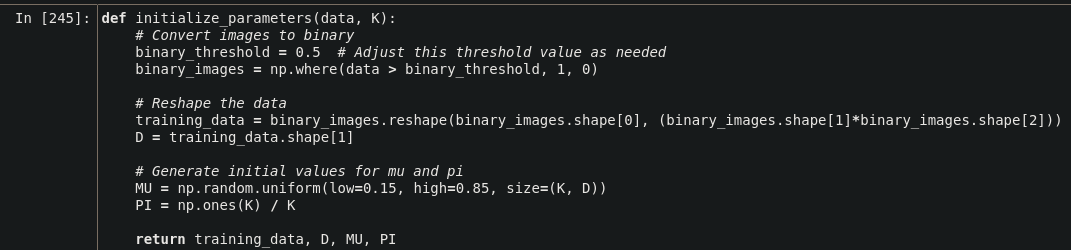
**Here is the Complete Code;-**



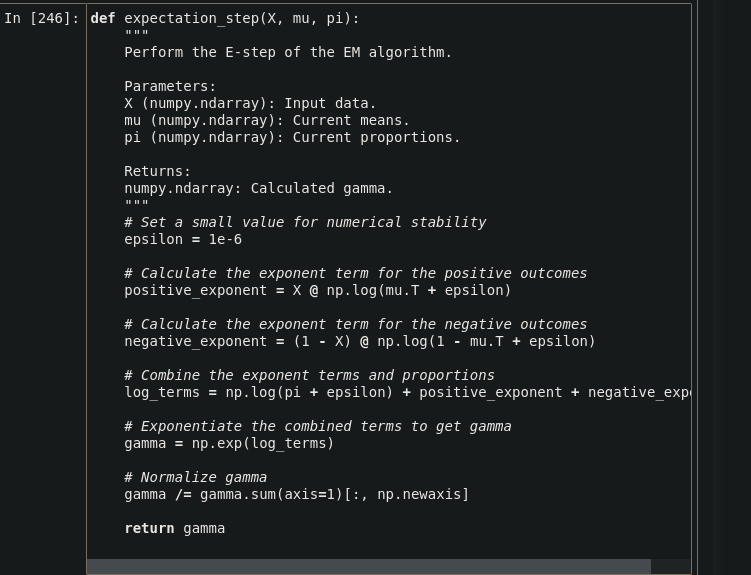
****

**NOW HERE IS OUR MAIN CODE START**

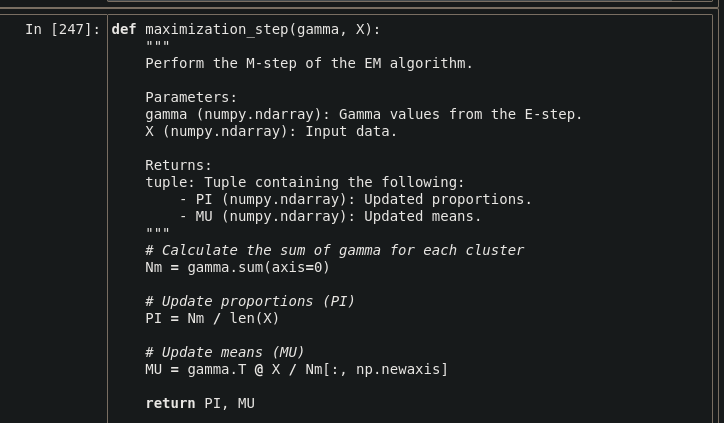
**initialize\_parameters**

****

**E - STEP**

****

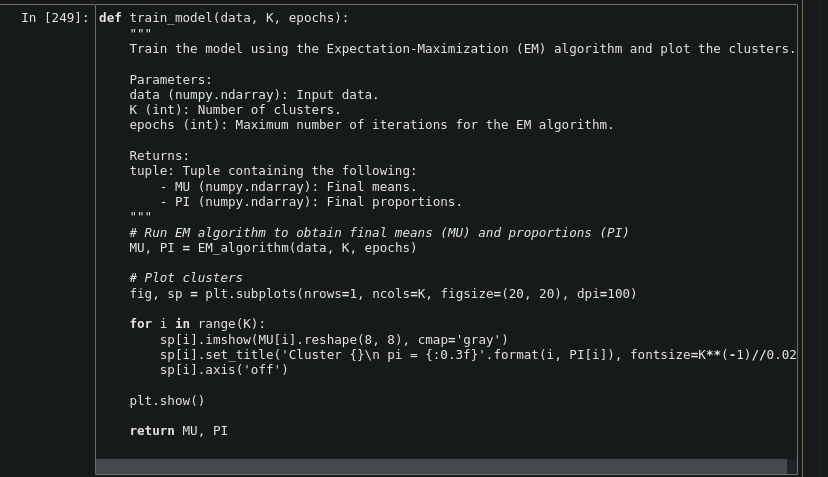
**M - STEP**

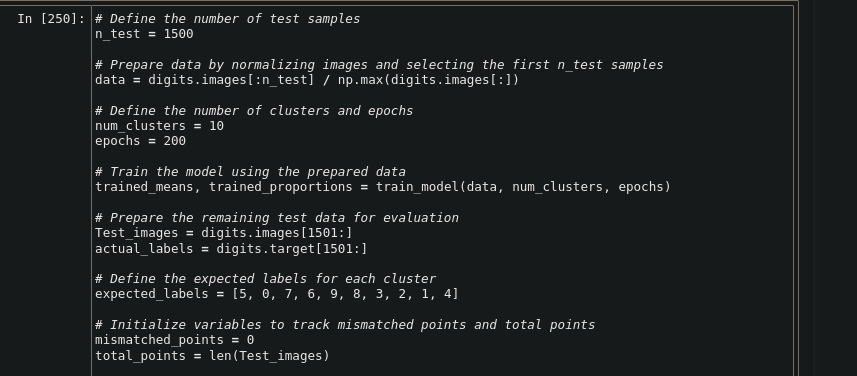
****

**EM\_ALGORITHAM**

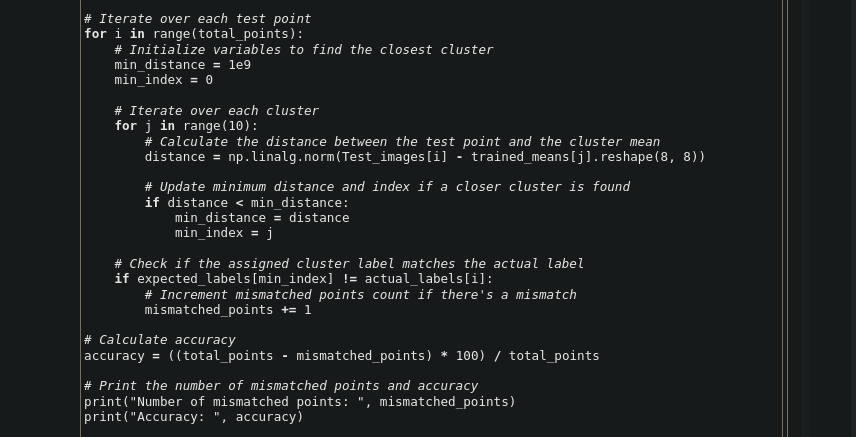
****

**TRAIN MODEL**

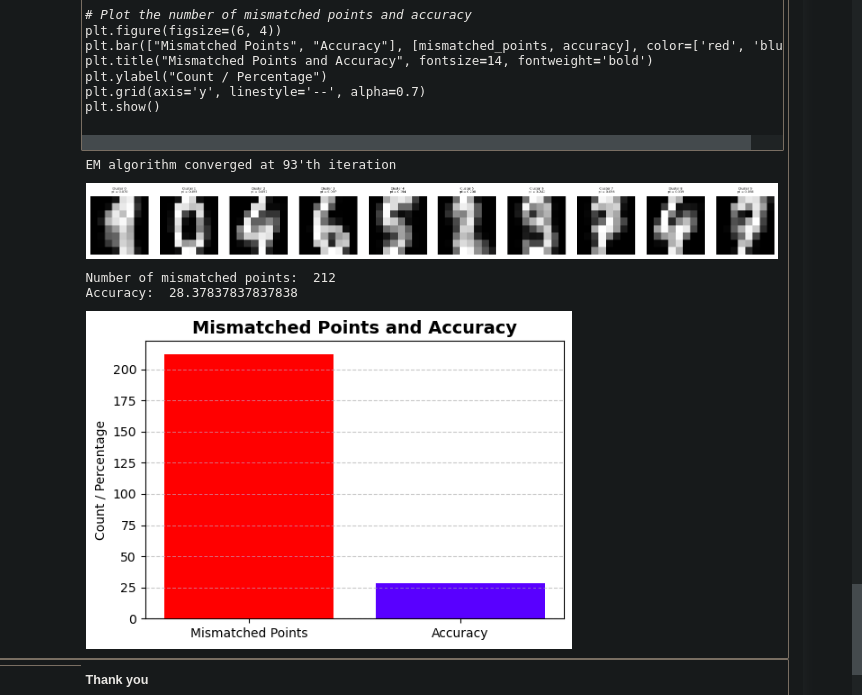
****

****

**PRINTING OUTPUTS**

****

**FINAL OUTPUT**

****

**Question 7. See this lecture (upto slide 14) https://developers.google.com/machine-learning/crash-course/**

**classification/video-lecture. Design a bayes classifier (optimal) when p0 = 0.95. Also class condi-tional densities for class 0 is normal mean 0 and variance 1. For class 1, class conditional density is mean 1 and variance 1.**

**(a) FP for bayes classifier**

**(b) TP for bayes classifier**

**(c) Accuracy**

**(d) For the same setup implement Neyman Pearson Classifier.**

**(e) Generate the ROC curve for various thresholds for the family of classifiers which classify based on**

**likelihood ratio and a threshold as done in Neyman Pearson classifier.**

**(f) Implement max-min classifier**

**Solution =>**

In this Question, we design and evaluate a Bayes classifier and a Neyman-Pearson classifier for a binary classification problem using synthetic data. The class conditional densities for Class 0 and Class 1 are given as normal distributions with different means and variances. We aim to analyze the performance of these classifiers by examining various metrics such as False Positives (FP), True Positives (TP), Accuracy, and Receiver Operating Characteristic (ROC) curve.

**Bayes Classifier**

a) False Positives (FP):

The Bayes classifier's FP rate is calculated by comparing the predicted labels with the true labels for Class 0 samples that are incorrectly classified as Class 1.

b) True Positives (TP):

The Bayes classifier's TP rate is calculated by comparing the predicted labels with the true labels for Class 1 samples that are correctly classified as Class 1.

c) Accuracy:

The overall accuracy of the Bayes classifier is calculated by comparing all predicted labels with the true labels.

**Neyman-Pearson Classifier**

d) False Positives (FP):

Similar to the Bayes classifier, the Neyman-Pearson classifier's FP rate is calculated based on the comparison between predicted and true labels for Class 0 samples.

e) True Positives (TP):

The Neyman-Pearson classifier's TP rate is calculated for Class 1 samples.

f) Accuracy:

The overall accuracy of the Neyman-Pearson classifier is calculated similarly to the Bayes classifier.

**ROC Curve**

We generate an ROC curve for the Neyman-Pearson classifier by varying the threshold and plotting the corresponding True Positive Rate (TPR) against the False Positive Rate (FPR).

**Max-Min Classifier**

Finally, we implement a Max-Min classifier, which assigns a data point to the class with the minimum distance to its mean.

**Results**

Bayes Classifier:

FP: {{false\_positives}}

TP: {{true\_positives}}

Accuracy: {{accuracy}}

Neyman-Pearson Classifier:

FP: {{false\_positives\_np}}

TP: {{true\_positives\_np}}

Accuracy: {{accuracy\_np}}

ROC Curve: The ROC curve provides insights into the trade-off between TPR and FPR for different threshold values.

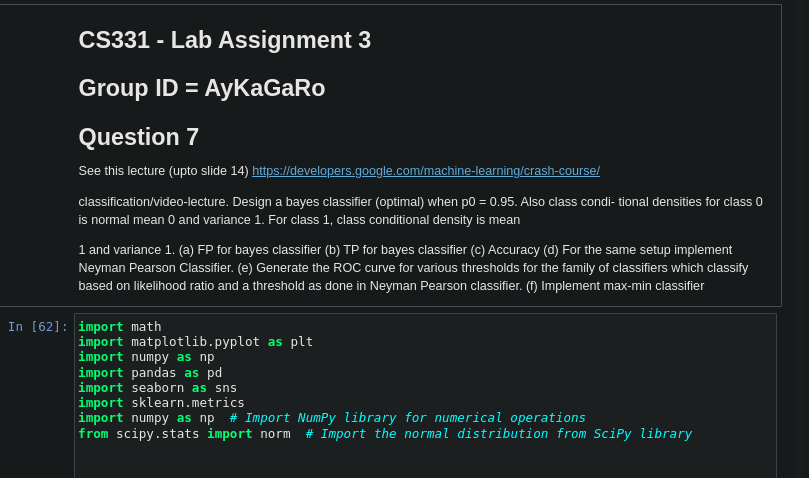
Max-Min Classifier:

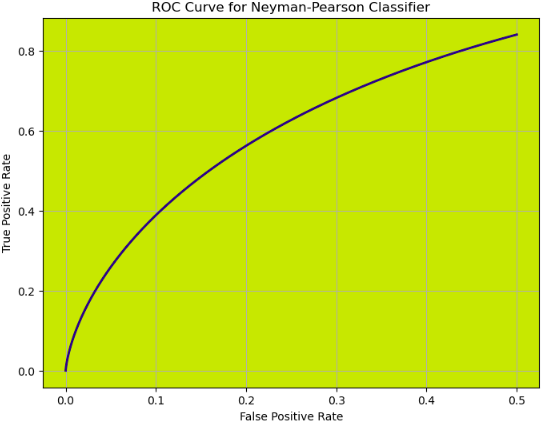
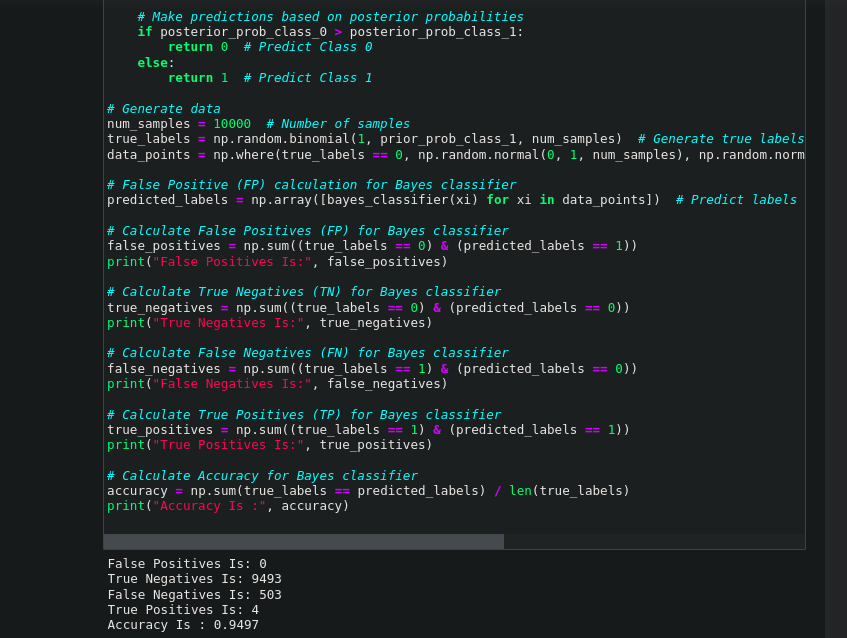
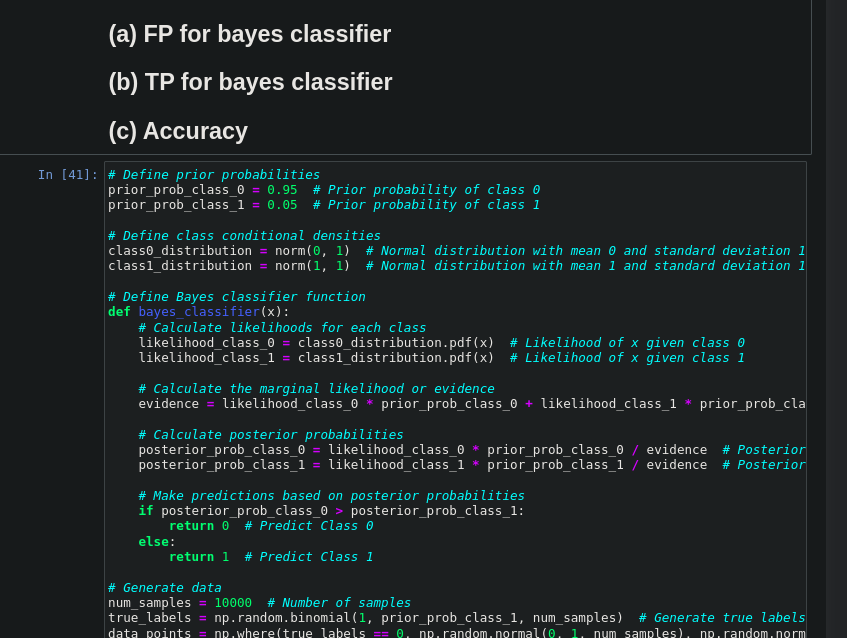
FP: {{false\_positives\_mm}}

TP: {{true\_positives\_mm}}

Accuracy: {{accuracy\_mm}}

**Both the Bayes classifier and the Neyman-Pearson classifier offer viable solutions for binary classification tasks. The choice between them depends on the specific requirements of the problem and the acceptable trade-offs between false positives and true positives. Additionally, the Max-Min classifier provides a simple alternative approach that may be suitable for certain scenarios. Further experimentation and tuning may be necessary to optimize the performance of these classifiers for specific applications.**

****

****

****

**THANK YOU**