Deepam Sarmah
2020050
deepam20050@iiitd.ac.in

Assumptions:

1. The MNIST dataset is taken in the form of a csv format

Approach:

1. I start by loading the dataset in csv format using the pandas library
2. I then visualize the dataset by showing 5 samples of each digit 0 to 9.
3. I then generate the mean vector for all the labels using numpy.mean() method.
4. I compute the covariance matrix by using the formula $\frac{1}{N-1} \cdot \sum_{i=1}^{N} (X_i - \overline{X})(X_i - \overline{X})^T$.
5. I apply regularization to overcome the issue of covariance matrix being singular i.e, if Σ be the covariance matrix then I do $\Sigma \leftarrow \Sigma + \lambda I$ where I is the identity matrix and λ is the regularization factor which can be a very small value such as 10^{-6}.
6. We use N - 1 and not N because we want to get an unbiased estimate of the covariance.
7. I create my own accuracy function which computes the number of matching terms in the predicted labels array and the actual test labels divided by the total number of test labels.
8. For LDA from scratch:
    a. I generate the weighted covariance matrix $\Sigma = \frac{n_1\Sigma_1 + n_2\Sigma_2 + \ldots + n_d\Sigma_d}{n_1 + n_2 + \ldots + n_d}$ and use it in the linear discriminant analysis formula.
    b. I use the LDA formula taught in class to find the discriminants $g_i(x)$ for all the labels
    c. I then assign $arg\ max\ (i)\ g_i(x)$ to be the classification of the x
9. For QDA from scratch:
    a. I use the covariance matrices which I computed in step 4 and use the formula given in the classroom to compute the discriminant $g_i(x)$ for each label.
    b. I then assign $arg\ max\ (i)\ g_i(x)$ to be the classification of the x
10. For LDA and QDA sklearn: I fit the model with the training i.e, mnist_train.csv dataset and then compute the predicted values on the mnist_test.csv dataset. I compare the predicted values with the actual values and compute the accuracy of sklearn LDA and QDA

Results:

Scratch LDA Accuracy = 0.86
Scratch QDA Accuracy = 0.5290
sk-learn LDA Accuracy = 0.873
/home/deepam/.local/lib/python3.10/site-packages/sklearn/discriminant_analysis.py:926:
UserWarning: Variables are collinear
warnings.warn("Variables are collinear")
sk-learn QDA Accuracy = 0.5384

We see that the results from the scratch's and the sklearn's LDA and QDA are similar with only a few precision errors. By doing multiple runs we can see that the scratch and sklearn implementation both have similar accuracies.

References:

1. https://www.kaggle.com/learn/pandas