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. I add random gaussian noise to the dataset and also shuffle the rows of the training and testing dataset.
2. I then visualize the dataset by showing 5 samples of each digit 0 to 1.
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 $\Sigma = \frac{n_1\Sigma_1 + n_2\Sigma_2 + \dots + n_d\Sigma_d}{n_1 + n_2 + \dots + n_d}$ 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 PCA from scratch:
    a. I centralize the data matrix along the mean.
    b. I then compute the covariance matrix Σ.
    c. For this covariance matrix I compute its eigenvalues and eigenvectors and depending upon n get the largest n eigenvectors sorted by their eigenvalues.
    d. These n sorted eigenvectors would be the directions along which the data matrix would be projected along. These n sorted eigenvectors form the PCA matrix.
    e. Then for reconstruction I use the PCA matrix and multiply it with X. To this I add the original data matrix's mean to arrive at the original data matrix.
    f. For reconstruction error I find the absolute difference between the original and projected data and plot it.
10. For FDA from scratch:
    a. I get the mean vector of the data matrices.
    b. I then find the covariance matrices from the data matrix.
    c. I compute the within class scatter S_w = S_1 + S_2

d. I compute w using the formula $w = S_w(\overline{\mu_1} - \overline{\mu_2})$ as there are only 2 categories.

11. For each subtask that requires me to use FDA, PCA and LDA I invoke the required function call.

Results:

```
[PCA] n =  2 => Scratch LDA Accuracy =  0.7650118203309693
[PCA] n =  3 => Scratch LDA Accuracy =  0.7829787234042553
[PCA] n =  5 => Scratch LDA Accuracy =  0.8156028368794326
[PCA] n =  8 => Scratch LDA Accuracy =  0.9267139479905437
[PCA] n =  10 => Scratch LDA Accuracy =  0.9356973995271868
[PCA] n =  15 => Scratch LDA Accuracy =  0.9536643026004729
[FDA]  => Scratch LDA Accuracy =  0.9976359338061466
[PCA + FDA + LCA] n =  2 => Scratch LDA Accuracy =  0.7635933806146572
[PCA + FDA + LCA] n =  3 => Scratch LDA Accuracy =  0.7829787234042553
[PCA + FDA + LCA] n =  5 => Scratch LDA Accuracy =  0.8170212765957446
[PCA + FDA + LCA] n =  8 => Scratch LDA Accuracy =  0.9267139479905437
[PCA + FDA + LCA] n =  10 => Scratch LDA Accuracy =  0.9356973995271868
[PCA + FDA + LCA] n =  15 => Scratch LDA Accuracy =  0.9536643026004729
```

References:

1. https://www.kaggle.com/learn/pandas
2. https://www.amazon.in/Introduction-Machine-Learning-Python-Scientists/dp/9352134575