Study on Dimensionality Reduction for Classification

Zuo Jia
JZUO001@e.ntu.edu.sg
School of Electrical and Electrical Engineering
Nanyang Technological University

***Abstract*—Dimensionality reduction place an important role in helping enhance the classification performance. High dimensional data often presents great difficulties for robust and accurate identification. In this study, I focus on two datasets: the handwritten digits dataset and the CIFAR-10 dataset. I investigate two dimensionality reduction methods, namely Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), and employ the Mahalanobis distance classifier for classification.**

My work can be primarily divided into three aspects: firstly, I experimentally explore the relationship between dimensionality reduction and classification accuracy on both datasets. Next, I compare the impact of PCA and LDA on classification performance. Finally, I investigate the range of values for the regularization parameter of the covariance matrix.

***Index Terms*—PCA, LDA, Mahalanobis distance classifier, regularization parameter**

I. Introduction

High-dimensional data are ubiquitous in real-world applications such as image, video, text and audio data. Classification of such high-dimensional data suffers from the curse of dimensionality, where inclusion of redundant features leads to inaccurate classification. Dimensionality reduction is thus imperative as a pre-processing step before feeding the data to a classifier. Principal component analysis (PCA) and linear discriminant analysis (LDA) have emerged as two popular techniques for dimensionality reduction. However, the choice of technique impacts the classification performance differently and the final reduced dimension also plays a crucial role.

In this work, I empirically evaluate the effects of PCA and LDA on the classification accuracy using two distinct high-dimensional datasets. The raw datasets are first reduced to lower dimensions separately using PCA and LDA, before being classified using the minimum Mahalanobis distance classifier. By analyzing the experimental results and drawing connections to theory, I aim to provide novel insights into the working mechanics of the two techniques. The findings from this comparative study shall guide the selection of suitable dimensionality reduction solutions for real-world pattern classification problems.

II. Related works

A. Dimensionality Reduction methods

High dimensional data is ubiquitous in scientific research and industrial production. Different dimensionality reduction methods are used to reduce the dimension of the data. Dimensionality reduction methods can be classified into feature selection, projection method, sparse learning, and kernel method etc. [1]. Two common linear dimensionality reduction methods such as LDA belong to kernel method, and PCA belongs to projection method. There are some nonlinear dimensionality reduction methods [2] such as non-negative matrix factorization (NMF) and kernel-based principal component analysis (KPCA).

B. Application of PCA and LDA in Classification

When the data dimensionality increases, the performance of the classifier improves. However, when the data dimensionality continues to increase, the performance of the classifier deteriorates. Many researchers have conducted research on how to apply appropriate dimensionality reduction methods before classification to improve classification accuracy.

A. Malhi et al. proposed a feature extraction scheme based on PCA method [3] to reduce dimensionality, which achieved more accurate defect classification with fewer feature inputs. Mia Hubert et al. proposed a robust PCA (ROBPCA) method [4] for classification problems in biosciences, which achieved good results. Another paper [5] studied PCA and its several linear and nonlinear variants, exploring their effects on hyperspectral remote sensing image (HSIs) classification, such as Segmented-PCA (SPCA), Spectrally Segmented-PCA (SSPCA), and kernel Entropy Component Analysis (KECA). One study [6] applied LDA to text classification and compared it with other well-known semi-supervised methods, demonstrating its usefulness for text classification tasks.

One paper [7] used PCA, LDA, and Independent Component Analysis (ICA) methods to reduce the dimensionality of electroencephalogram (EEG) signals, which were then used as inputs to a support vector machine (SVM). The effects of different methods on classification performance were compared. Another paper [8] applied PCA, LDA, and ICA to dimensionality reduction of electrocardiogram (ECG) signals separately, and input the reduced data into different classifiers to analyze the advantages and disadvantages of different dimensionality reduction methods.

A study [9] explores the use of Asymmetric Principal Component Analysis(AsymPCA) and Asymmetric Linear Discriminant Analysis(AsymLDA) for pattern classification, with a focus on addressing data distribution asymmetry to improve classification performance. It is expected to provide a novel approach to handling asymmetric data in classification tasks.

III. METHODOLOGY

A. Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a popular unsupervised learning algorithm that aims to find the directions of maximum variance in the data. It creates new features based on linear combinations of the original ones through the eigenvectors of the covariance matrix.

Given a dataset matrix X of dimensions $m \times n$, where m is the number of samples, n is the number of features. Firstly, zero-center each feature:

$$x^i = x_i - mean(x_i) \tag{1}$$

Then the covariance matrix S can be computed as:

$$S = \frac{1}{m} \times X^T \times X \tag{2}$$

where X^T is the transpose of the matrix X. The eigenvalues diagonal matrix D and corresponding eigenvectors matrix V of the covariance matrix S can be calculated by:

$$S \times V = D \times V \tag{3}$$

The projection matrix W is formed by selecting the eigenvector corresponding to the largest k eigenvalues:

$$W = V(:, 1 : k) \tag{4}$$

Then, project the original data onto W to obtain the reduced data Y:

$$Y = X \times W \tag{5}$$

By projecting through PCA, high-dimensional data can be mapped to low-dimensional space while retaining the most important feature information.

B. Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) is a classical supervised learning dimensionality reduction method. It aims to find a projection matrix W, which used to map the data in high dimension to low dimension. It makes the projection points of the same kind of samples as close as possible, and the projection points of different kinds of samples as far as possible.

Given a dataset matrix X. K is the number of classes. Firstly, calculate the mean of every class μ_c where $c = 1, 2...k$ is the c-th class.

Then calculate the within-class scatter matrix S_W and the between-class scatter matrix S_B by:

$$S_W = \sum_{c=1}^{K} \sum_{x \in K_c} (x - \mu_c)(x - \mu_c)^T \tag{6}$$

$$S_B = \sum_{c=1}^{K} n_c(\mu_c - \mu)(\mu_c - \mu)^T \tag{7}$$

where μ is the overall mean vector, n_i is the number of samples in the c-th class.

Calculate the eigenvalues diagonal matrix D and corresponding eigenvectors matrix V of $S_W^{-1} \times S_B$ by solving:

$$S_W^{-1} \times S_B \times V = D \times V \tag{8}$$

The eigenvectors corresponding to the largest k eigenvalues are selected to form the projection matrix W. Then, project the original data onto W to obtain the reduced data Y:

$$Y = X \times W \tag{9}$$

Then we can get the data in low-dimensional space.

C. Mahalanobis Distance Classifier

The Mahalanobis distance classifier is a common classification algorithm that uses the Mahalanobis distance between samples and the center of each class to classify them.

Given K different classes. The covariance matrix of the k-th class is S_k. The Mahalanobis distance D_k between a data point x and the k-th class mean μ_k can be computed as:

$$D(x, \mu_k) = \sqrt{(x - \mu_k)^T \times S_k^{-1} \times (x - \mu_k)} \tag{10}$$

For every single data point x, compute the Mahalanobis distance between x and the center of every class. Then classify the data point to the class with the smallest Mahalanobis distance.

To further improve classification accuracy, regularization of the covariance matrix is necessary. A common approach [10] is to add a constant to the diagonal elements of the covariance matrixS_k:

$$D(x, \mu_k) = \sqrt{(x - \mu_k)^T \times (S_k^{-1} + aI) \times (x - \mu_k)} \tag{11}$$

where a is the regularization parameter that controls the strength of regularization, balancing between the original covariance matrix and diagonal approximation. When a takes a small value, it is equivalent to no regularization, with distance calculation using the original covariance matrix. When a takes a large value, it approximates the covariance matrix with a diagonal matrix, ignoring correlations between features and only considering the variance of each feature. Appropriate regularization can improve numerical stability and predictive performance of the algorithm.

IV. EXPERIMENT

In this section, I first selected two different datasets and performed corresponding preprocessing on each dataset respectively. Next, experiments were conducted to investigate the relationship between dimensionality and classification accuracy. In the third part, dimensionality reduction experiments using PCA and LDA were carried out respectively, comparing the strengths and weaknesses of both methods. Finally, experiments were conducted to investigate the effect of the regularization parameter on classification performance.

A. Dataset Preparation

1) Handwritten Digit Dataset: I utilized a handwritten digit dataset containing 60000 training images and 10000 test images. Each image was a 28 $\times$ 28 pixel grayscale image, yielding 784 dimensions total. This dataset can be applied to a 10-class classification task. As depicted in Figure 1, example images from the handwritten digit dataset are shown.

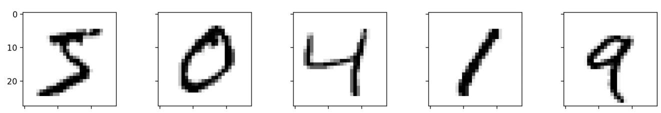

Fig. 1. Example Images from the Handwritten Digit Dataset

2) CIFAR-10 Dataset: The CIFAR-10 dataset consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. Among these, 50,000 images are in the training set and 10,000 are in the test set. As depicted in Figure 2, example images from CIFAR-10 dataset are shown.

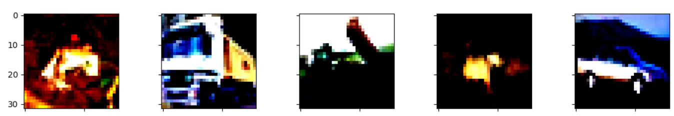

Fig. 2. Example Images from CIFAR-10 Dataset

3) Dataset Preprocessing: Firstly, the images were normalized and centered to accelerate model training and improve classification accuracy. Then, the images were reshaped into corresponding one-dimensional vectors for ease of input into the model.

B. Relationship Between Dimensionality and Classification Accuracy

1) Handwritten Digit Dataset: To investigate the relationship between dimensionality and classification accuracy, I first performed dimensionality reduction on the original data using PCA. I then trained a Mahalanobis distance classifier on the handwritten digit dataset's training set, and tested the trained model on the test set.

As shown in Figure 3, classification accuracy rose sharply between dimensions 1 to 10. After dimensionality surpassed 10, the rate of increase in classification accuracy slowed progressively.

This phenomenon can be attributed to the 10 classes in the handwritten digit dataset. When dimensionality was less than 10, insufficient dimensions failed to capture adequate features, resulting in severe information loss and poor performance. However, once dimensionality rose above 10, useful information could be sufficiently retained to enhance classification efficacy.

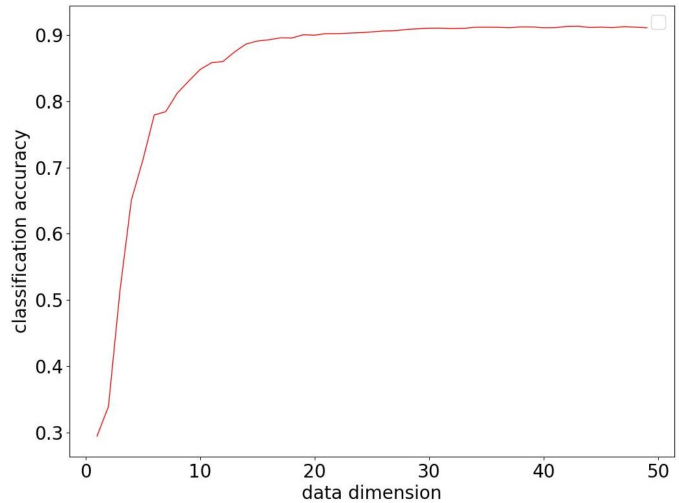

Fig. 3. Relationship Between Dimensionality and Classification Accuracy on Handwritten Digit Dataset

2) CIFAR-10 Dataset: Similar experiments investigating the relationship between dimensionality and classification accuracy were conducted using the CIFAR-10 dataset. As depicted in Figure 4, classification accuracy rose sharply between dimensions 1 to 10, then progressively slowed as dimensionality increased from 10 to 30. Beyond 30 dimensions, a noticeable decrease in classification accuracy was observed.

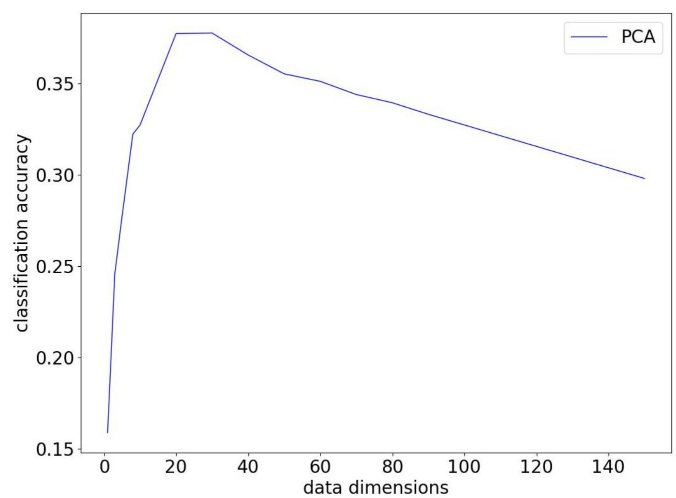

Fig. 4. Relationship Between Dimensionality and Classification Accuracy on CIFAR-10 Dataset

This phenomenon is interesting, with only 10 classes in the dataset, classification accuracy improved as dimensionality rose from 1 to 10. However, further increasing dimensionality beyond 10 retained excessive redundant and noisy information. Such extraneous information interferes with the classification model, resulting in overfitting and performance decline. And this phenomenon is so called ”curse of dimensionality”.

To better understand the curse of dimensionality, I plotted the variance explained by the top 100 principal components of the covariance matrix, as shown in Figure 5. It can be observed that the first 10 components accounted for the vast majority of the variance. As dimensionality increased, later components contained very little useful information, perfectly explaining the curse of dimensionality phenomenon.

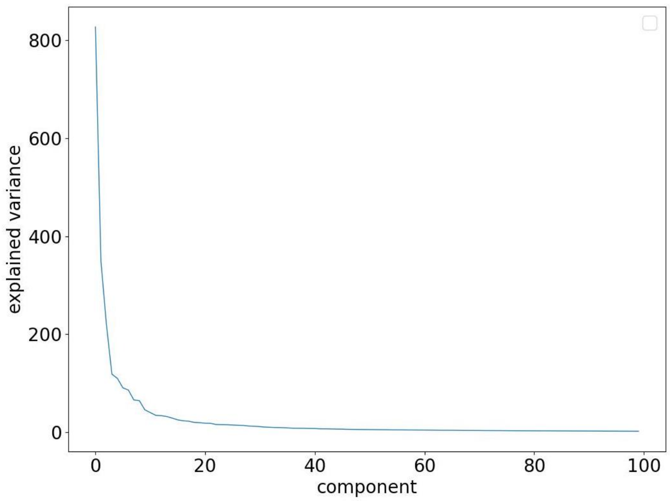

Fig. 5. Variance Explained by Top 100 Principal Components of the Covariance Matrix

The handwritten digit dataset did not exhibit the curse of dimensionality because the classification task is relatively simple, with clear differences between the 10 digit classes. The original high dimensional data did not contain excessive complex redundant or noisy features. In contrast, the CIFAR-10 dataset has substantial noise and redundancy, making removal of higher dimensions necessary.

C. Effect of PCA and LDA on Classification Performance

In this section, dimensionality reduction using LDA and PCA was performed on the handwritten digit dataset, followed by classification using a Mahalanobis distance classifier.

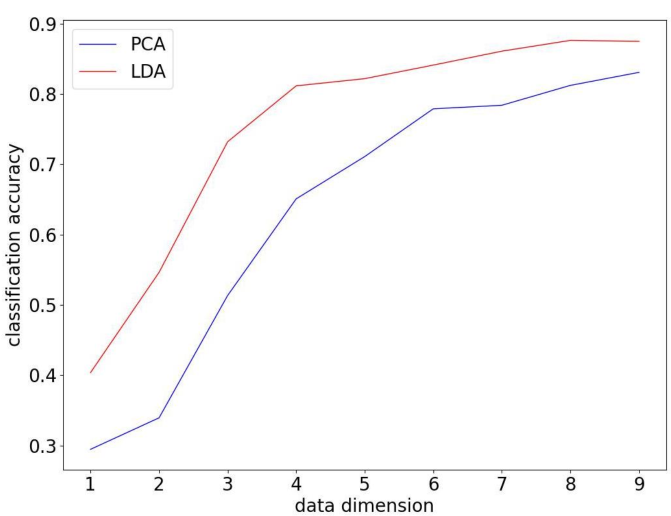

Fig. 6. Effect of PCA and LDA on Classification Performance

As depicted in Figure 6, it can be observed that classification accuracy was much higher using LDA dimensionality reduction compared to PCA. This is because PCA aims to retain the low-dimensional subspace with maximum total sample variance, without considering class information. In contrast, LDA aims to maximize class separability in the subspace. [11] Thus, LDA retains discriminative information between classes, while PCA preserves total variation but dilutes differences between classes. Since the goal of classification is distinguishing between classes, the LDA-reduced features are more advantageous.

Table 1 illustrates that within a range of 10 degrees, LDA outperforms PCA. However, it should be noted that LDA can only reduce the dimensionality to one less than the number of categories, whereas PCA allows for further dimensionality increase, potentially leading to better results.

TABLE I
COMPARISON OF THE EFFECTS OF PCA AND LDA ON CLASSIFICATION PERFORMANCE

Dimension	Accuracy with PCA	Accuracy with LDA
1	0.2946	0.4036
5	0.7109	0.8216
9	0.8302	0.8747
40	0.9106	-
Maximum Performance	0.9106	0.8747

In summary, PCA is an unsupervised dimensionality reduction method without utilizing class labels, while LDA is a supervised approach that identifies a subspace more effective for classification using this additional information. Therefore, the LDA-reduced features are more optimized for the classification task, greatly improving accuracy.

D. Effect of the Regularization Parameter on Classification Performance

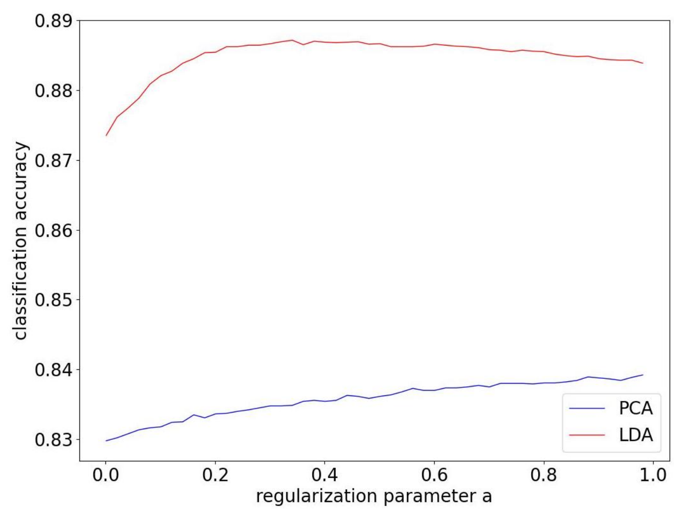

Fig. 7. Effect of the Regularization Parameter on Classification Performance

When using Mahalanobis distance classification, regularizing the covariance matrix can effectively improve classification accuracy. Therefore, the Mahalanobis distance calculation was modified using Equation 11. The value of the regularization parameter a is crucial. In this section, with dimensionality fixed at 9, regularization was performed on the covariance matrices obtained after PCA and LDA dimensionality reduction. Mahalanobis distance classification was then conducted to investigate the effect of the regularization parameter a on classification performance.

As shown in Figure 7, where a takes on values from 0.001 to 0.981, it can be observed that classification accuracy initially improves then slowly declines as the regularization parameter is increased. This phenomenon can be explained as follows:

(1) When a is very small, correlation information between different features in the covariance matrix is fully utilized, but invertibility and numerical stability of the covariance matrix are poor. This leads to inaccurate distance calculations and degraded classification performance.

(2) As a increases, invertibility and numerical stability of the covariance matrix gradually improve, enabling more accurate distance calculations and improved classification performance.

(3) However, as a continues to increase, the covariance matrix becomes over-regularized, losing inter-feature correlation information by using a diagonal approximation instead of the original covariance matrix. This introduces bias into distance calculations and reduces classification performance.

(4) When a is very large, correlations are completely ignored, relying solely on variance of individual features. Failure to fully leverage correlational data results in decreased classification capability.

V. Conclusion

In this study, I primarily investigated the impact of dimensionality reduction methods on classification algorithms. The major process can be summarized as follows: (1) Two different kind of datasets was selected and preprocessed. (2) The effect of dimensionality on classification accuracy was explored on two datasets respectively. (3) Dimensionality reduction via PCA and LDA was compared in terms of classification performance. (4) The impact of the regularization parameter value was examined.

This study demonstrates that when the dimensionality is less than the number of categories in the dataset, higher dimensions lead to improved classification performance, as they encompass more information. However, when the dimensionality exceeds the number of categories in the dataset, redundant noisy information is incorporated, resulting in decreased classification performance. This phenomenon is so called "curse of dimensionality".

Furthermore, I compared the impact of PCA and LDA on the classification performance of the Mahalanobis distance classifier. Experimental results indicate a clear superiority of LDA over PCA.

Finally, I conducted experiments to investigate the influence of the regularization parameter on the classification performance within the covariance matrix. The results indicate that there exists an optimal range for selecting the regularization parameter.

References

[1] X. Huang, L. Wu, and Y. Ye, "A review on dimensionality reduction techniques," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 33, no. 10, p. 1950017, 2019.

[2] W. Jia, M. Sun, J. Lian, and S. Hou, "Feature dimensionality reduction: a review," *Complex & Intelligent Systems*, vol. 8, no. 3, pp. 2663–2693, 2022.

[3] A. Malhi and R. Gao, "Pca-based feature selection scheme for machine defect classification," *IEEE Transactions on Instrumentation and Measurement*, vol. 53, no. 6, pp. 1517–1525, 2004.

[4] M. Hubert and S. Engelen, "Robust pca and classification in biosciences," *Bioinformatics*, vol. 20, no. 11, pp. 1728–1736, 2004.

[5] M. P. Uddin, M. A. Mamun, and M. A. Hossain, "Pca-based feature reduction for hyperspectral remote sensing image classification," *IETE Technical Review*, vol. 38, no. 4, pp. 377–396, 2021.

[6] M. Pavlinek and V. Podgorelec, "Text classification method based on self-training and lda topic models," *Expert Systems with Applications*, vol. 80, pp. 83–93, 2017.

[7] A. Subasi and M. I. Gursoy, "Eeg signal classification using pca, ica, lda and support vector machines," *Expert systems with applications*, vol. 37, no. 12, pp. 8659–8666, 2010.

[8] R. J. Martis, U. R. Acharya, and L. C. Min, "Ecg beat classification using pca, lda, ica and discrete wavelet transform," *Biomedical Signal Processing and Control*, vol. 8, no. 5, pp. 437–448, 2013.

[9] X. Jiang, "Asymmetric principal component and discriminant analyses for pattern classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 931–937, 2008.

[10] X. Jiang, B. Mandal, and A. Kot, "Eigenfeature regularization and extraction in face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 3, pp. 383–394, 2008.

[11] X. Jiang, "Linear subspace learning-based dimensionality reduction," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 16–26, 2011.

```python
import numpy as np
from sklearn.decomposition import PCA
from sklearn.discriminant_analysis import
    LinearDiscriminantAnalysis as LDA
from sklearn.metrics import accuracy_score
from sklearn.datasets import fetch_openml
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from keras.datasets import cifar10

# dataset selection: 1: CIFAR-10, 0: MNIST
dataset_selection = 1

if dataset_selection == 1:
    (X_train, y_train), (X_test, y_test) = cifar10.
        load_data()
    X_train = X_train.reshape(X_train.shape[0], -1)
    X_test = X_test.reshape(X_test.shape[0], -1)
    y_train = y_train.reshape(1, -1)
    y_test = y_test.reshape(1, -1)
    y_train = y_train[0]
    y_test = y_test[0]
    X_train = X_train.astype('float32')
    X_test = X_test.astype('float32')

elif dataset_selection == 0:
    mnist = fetch_openml("mnist_784", version=1)
    X, y = mnist["data"], mnist["target"]
    X, y = mnist.data.to_numpy(), mnist.target.
        to_numpy().astype(int)
    # split the dataset
    X_train, X_test, y_train, y_test =
        train_test_split(X, y, test_size=0.2,
        random_state=42)

# normalization and decentralization
mean = X_train.mean(axis=0)
std = X_train.std(axis=0)
X_train = (X_train - mean) / (std + 1e-8)
X_test = (X_test - mean) / (std + 1e-8)

# show 5 pictures
fig, axes = plt.subplots(nrows=2, ncols=5, sharex=
    True, sharey=True, figsize=(12,4))
for i in range(5):
    if dataset_selection == 1:
        img = X_train[i].reshape(32, 32, 3)
    elif dataset_selection == 0:
        img = X_train[i].reshape(28, 28)
    axes[0][i].imshow(img, cmap='Greys')
plt.tight_layout()
plt.show()

def lda_reduction(dimes):
    lda = LDA(n_components=dimes)
    lda.fit_transform(X_train,y_train)
    X_train_lda = lda.transform(X_train)
    X_test_lda = lda.transform(X_test)
    return X_train_lda, X_test_lda

# if show the Variance Explained of Covariance
    Matrix
show_explained_variance = 0

def pca_reduction(dimes):
    pca = PCA(n_components=dimes)
    X_train_pca = pca.fit_transform(X_train)
    X_test_pca = pca.transform(X_test)
    if show_explained_variance == 1:
        explained_variance = pca.explained_variance_
        print("Explained Variance:",
            explained_variance)
        plt.figure(figsize=(12, 9))
        plt.tick_params(labelsize = 20)
        plt.plot(range(len(explained_variance)),
            explained_variance, linewidth = 1)
        plt.legend(fontsize = 20)
        plt.xlabel('component', fontsize = 20)
        plt.ylabel('explained variance', fontsize =
            20)
        plt.savefig('explained_variance.jpg')
    return X_train_pca, X_test_pca

# dimensions: list of dimensions
# reduction: choose LDA(0) or PCA(1)
# a_: regularization parameter a
def classify(dimensions, reduction, a_):
    accuracies = []
    for n_components in dimensions:
        if reduction == 0: # lda
            X_train, X_test = lda_reduction(
                n_components)
        elif reduction == 1: # pca
            X_train, X_test = pca_reduction(
                n_components)

        # train classifier
        unique_classes = np.unique(y_train)
        classifier = {}
        for c in unique_classes:
            X_train_class = X_train[y_train == c]
            classifier[c] = {"mean": X_train_class.
                mean(axis=0), "cov": np.cov(
                X_train_class, rowvar=False)}
            # regularized the covariance matrix
            cov_dimension = X_train_class.shape[1] #
                 get the dimension of cov
            classifier[c]["cov"] = classifier[c]["
                cov"] + a_ * np.eye(cov_dimension)

        # test classifier
        predictions = []
        for x in X_test:
            distances = [np.sqrt((x - classifier[c][
                "mean"]).T @ np.linalg.inv(
                classifier[c]["cov"]) @ (x -
                classifier[c]["mean"])) for c in
                unique_classes]
            predicted_class = unique_classes[np.
                argmin(distances)]
            predictions.append(predicted_class)

        accuracy = accuracy_score(y_test,
            predictions)
        accuracies.append(accuracy)
        print(f"Dimension: {n_components}, Accuracy:
             {accuracy}")
    return accuracies

# main
dimensions = list(range(1, 10))
accuracies_pca = classify(dimensions, 1, 0.02)
accuracies_lda = classify(dimensions, 0, 0.02)

plt.figure(figsize=(12, 9))
plt.tick_params(labelsize = 20)
plt.plot(dimensions, accuracies_pca, color = 'blue',
     linewidth = 1, label = 'PCA')
plt.plot(dimensions, accuracies_lda, color = 'red',
    linewidth = 1, label = 'LDA')
plt.legend(fontsize = 20)
plt.xlabel('data dimensions', fontsize = 20)
plt.ylabel('classification accuracy', fontsize = 20)
plt.savefig('pca_lda_compare.jpg')
```