CPEN 355 Homework 4

Aloysio Kouzak Campos da Paz
Student ID 58687526

Collaboration statement I completed the assignment with the help of the internet and ChatGPT for understanding the concepts behind the assignment and for generating Python code.

Here is a link to the chat I had with ChatGPT:
https://chat.openai.com/share/3470ca47-befc-4ea4-94dd-ba6024aeb1a2

1 Coding Practice

We will use Principal Component Analysis (PCA) to reduce the dimension of images of handwritten digits, and then:

- Reconstruct the images to see how they look like
- Use Logistic Regression and the low dimensional data to classify the digits

We are using data of handwritten digits from the MNIST dataset, which contains 60,000 samples of handwritten digits. Each handwritten digit is an image of 28x28 pixels. So this dataset has shape (60000,28,28).

Above is an image showing a subsample of the MNIST dataset, taken from Wikipedia MNIST Database Wikipedia Page

1.1 Question 1.1: PCA for dimension reduction

We will reconstruct images using individual principal components to compare how much information is explained by each principal component.

When we do PCA, the principal components are ranked based on how much each component explains the data. We will use "k" to represent the principal component rank. So k = 0 explains most of the data, and larger k's explain less of the data.

We will reconstruct images using the following principal components, one at a time:

$$k = 0, 10, 20, 30, 40, 49 \tag{1}$$

We will use the following equation to reconstruct images

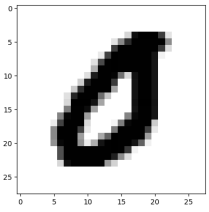
$$\hat{x}_i = \bar{X} + (\phi_k^\top x_i)\phi_k \tag{2}$$

Where $\hat{x}_i$ represents the reconstructed image, $\bar{X}$ represents the average image of all 0's, (ϕ_k) represents the kth principal component, and $(\phi_k^\top x_i)$ represents the projection of the original image on the kth principal component.

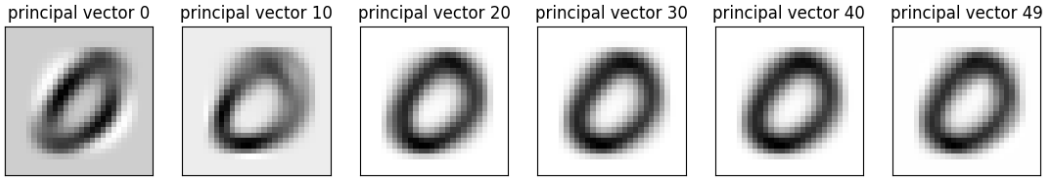
Here's pseudo-code describing the algorithm that we will follow:

```
Import PCA libray from sklearn

Compute the average image

Perform PCA on images of zeros using 50 principal components

Pick an image of a "0" to be reconstructed

Enter for loop to reconstruct images for each k:
    Reconstruct image using the k-th principal component
       through equation 2

Display the reconstructions for each k
```

Here's the random image we selected of a "0" from the MNIST dataset.

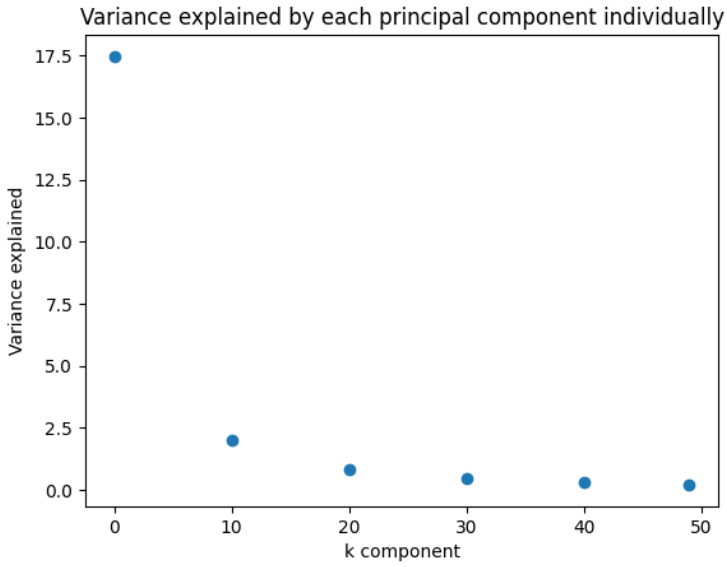


And here's how it's reconstruction looks like for each value principal component rank:



As we can see, when we use k = 0 the reconstructed image looks most similar to our original image. This is due to the first principal component explaining most of the variance in the data.

In the following graph of variance explained vs. k value we can observe that the larger the principal component, the less variance it explains.



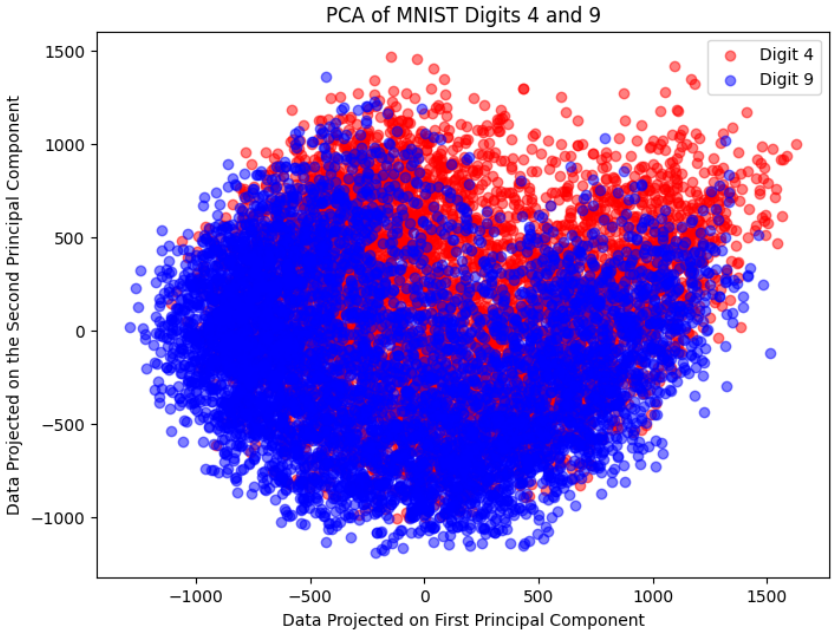
1.2 Question 1.2: PCA for classification

1.2.1 Comparing the first principal components of "4" and"9"

We will check how separable is the data when we use the first two principal components of numbers 4 and 9. We do this by plotting the data projected on the first component vs. the second component.

The algorithm used is like follows:

```
Load MNIST dataset

Flatten the image

Find indices where the label is either 4 or 9

Create the subset containing only images with labels 4 or 9

Perform PCA to reduce data to 2 dimensions

Extract the coordinates of the projected data

Plot
```



We observed each number as a dot on the scatter plot. As we can see, the scatter plot for the numbers 4 and 9 is not completely separable into two clusters. This is probably because 4 and 9 look alike with the closed loop on the top of the digit. Since the two clusters are not completely separable, training a machine learning algorithm to classify 4 and 9 could be hard.

1.2.2 Comparing the accuracy of classifying "4" and "9" as we use more principal components

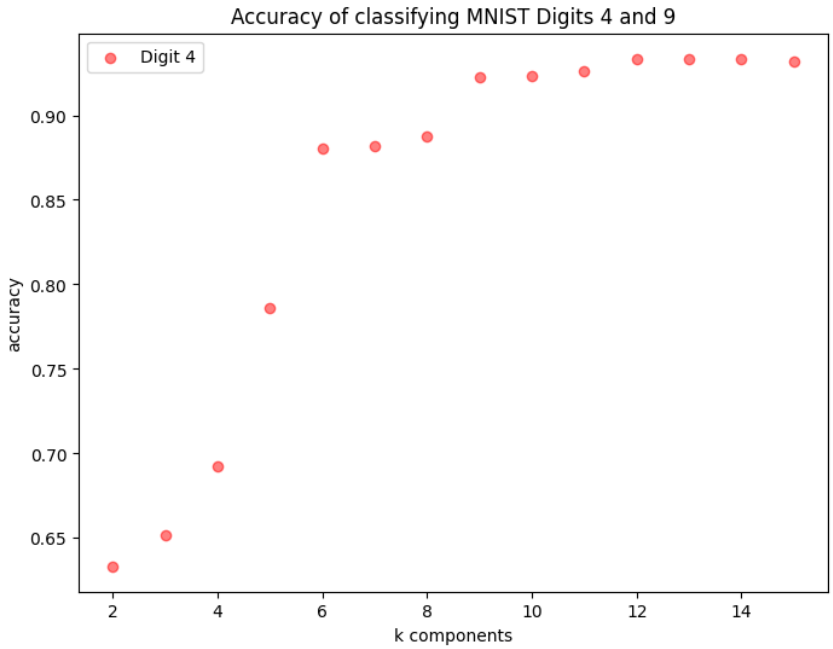
We will now analyze the effect of multiple principal components together by trying to classify the numbers "4" and "9' and comparing the classification accuracy. Note that in question 1.1 we used individual principal components, and now we use more than one together.

We will use logistic regression to classify the digits. The data we use to train the logistic regression will be the projection of the original data on the principal components.

This is the algorithm we follow:

```
Load the MNIST dataset

Flatten the images

Filter out all the images that are not 4 or 9

Encode the labels for binary classification: 4 becomes 0, 9
    becomes 1

Loop k_components in range (2,16):

    Perform PCA to reduce the data to 'k' principal components

    Split the data into training and testing sets

    Initialize the logistic regression model

    Train the model using the training data

    Predict on the test set

```

```
    Calculate the accuracy

    Print a detailed classification report

Plot accuracies vs. number of components used in PCA
```



We observe that for higher k values, the accuracy is higher. Then for values greater than k= 9, the accuracy stops increasing (plateaus). The plot makes sense, because when we use more principal components (higher k values) we are using more information to describe the numbers 4 and 9, which can make it easier to tell the numbers apart.

Furthermore, the accuracy stops increasing forever because at a certain point more information may not help to discern both numbers. In other words, the bottleneck may become something else, such as the limitations of our model (logistic regression).

2 Short Answer Questions

2.1 Question 2a: K-means clustering algorithm

Answer: 2 and 5.5

We will iterate the k-means algorithm until convergence. Our initial cluster centers are 1.8 and 2.8.

1. **Iteration 1:**
   - Distance from each number to each cluster center:

$$
\begin{aligned}
|1-1.8| &= 0.8 & |1-2.8| &= 1.8 \\
|2-1.8| &= 0.2 & |2-2.8| &= 0.8 \\
|3-1.8| &= 1.2 & |3-2.8| &= 0.2 \\
|4-1.8| &= 2.2 & |4-2.8| &= 1.2 \\
|5-1.8| &= 3.2 & |5-2.8| &= 2.2 \\
|6-1.8| &= 4.2 & |6-2.8| &= 3.2 \\
|7-1.8| &= 5.2 & |7-2.8| &= 4.2
\end{aligned}
$$

- Therefore, numbers 1 and 2 are closest to the cluster at 1.8, and numbers 3, 4, 5, 6, 7 are closest to the cluster at 2.8.
- Now we calculate the mean of each cluster:

$$\text{Mean of cluster } (1,2) = 1.5$$
$$\text{Mean of cluster } (3,4,5,6,7) = 5$$

- These means are the new cluster centers.

2. **Iteration 2:**
   - Distance from each number to each cluster center:

$$|1-1.5| = 0.5 \qquad |1-5| = 4$$
$$|2-1.5| = 0.5 \qquad |2-5| = 3$$
$$|3-1.5| = 1.5 \qquad |3-5| = 2$$
$$|4-1.5| = 2.5 \qquad |4-5| = 1$$
$$|5-1.5| = 3.5 \qquad |5-5| = 0$$
$$|6-1.5| = 4.5 \qquad |6-5| = 1$$
$$|7-1.5| = 5.5 \qquad |7-5| = 2$$

   - Numbers 1, 2, and 3 are closest to the cluster at 1.5; numbers 4, 5, 6, 7 are closest to the cluster at 5.
   - The new cluster centers are calculated as:

$$\frac{1+2+3}{3} = 2$$
$$\frac{4+5+6+7}{4} = 5.5$$

3. **Iteration 3:**
   - Distance from each number to each cluster center:

$$|1-2| = 1 \qquad |1-5.5| = 4.5$$
$$|2-2| = 0 \qquad |2-5.5| = 3.5$$
$$|3-2| = 1 \qquad |3-5.5| = 2.5$$
$$|4-2| = 2 \qquad |4-5.5| = 1.5$$
$$|5-2| = 3 \qquad |5-5.5| = 0.5$$
$$|6-2| = 4 \qquad |6-5.5| = 1.5$$
$$|7-2| = 5 \qquad |7-5.5| = 2.5$$

   - Numbers 1, 2, 3 continue to be closest to the cluster at 2; numbers 4, 5, 6, 7 continue to be closest to the cluster at 5.5.
   - The algorithm has converged, and we stop iterating.

2.2 Question 2b: statements about PCA

Answers: 1 TRUE, 2 FALSE, 3 TRUE, 4 FALSE