CpE 520: HW #3

West Virginia University

Ali Zafari

Contents

1 Preface

At first, we need to import the required modules and of course the MNIST dataset:

[1]:
```
import mnist
import os
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from scipy.linalg import eigh

mnist.temporary_dir = lambda: os.getcwd()

# Each of these functions first downloads the data and returns a numpy array.
train_images = mnist.train_images()
train_labels = mnist.train_labels()

assert train_images.shape == (60000, 28, 28), 'train_imgages shape is not␣
↪correct'
assert train_labels.shape == (60000,), 'train_labels shape is not correct'
```

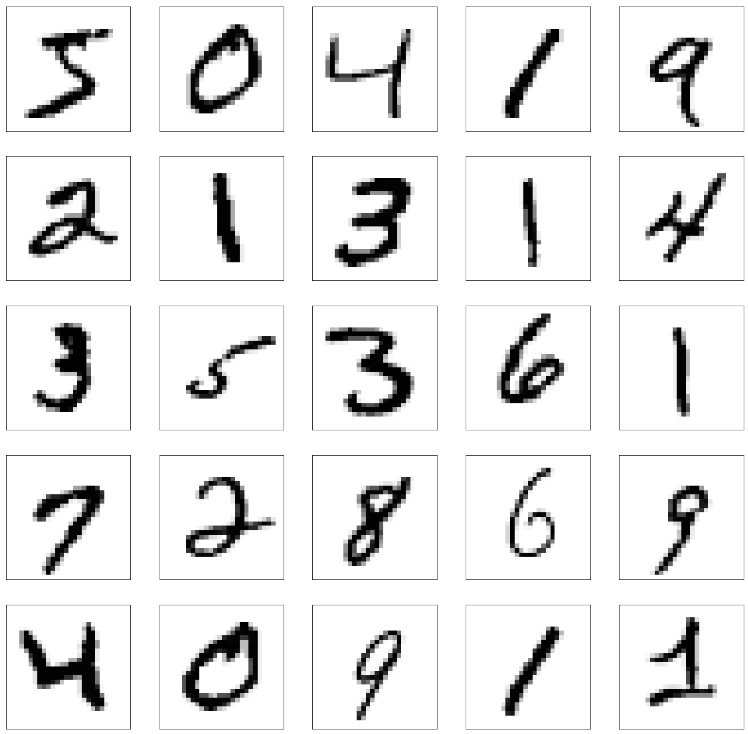
Finding how many of every digit exists in the mnist dataset:

[2]:
```
whole_number = 0

for i in range(10):
    number = train_images[train_labels == i, :].shape[0]
    whole_number += number
    print(f'number of digit {i} is: {number}')

assert whole_number == train_images.shape[0], 'some data is missed'
```

```
number of digit 0 is: 5923
number of digit 1 is: 6742
number of digit 2 is: 5958
number of digit 3 is: 6131
number of digit 4 is: 5842
number of digit 5 is: 5421
number of digit 6 is: 5918
number of digit 7 is: 6265
number of digit 8 is: 5851
number of digit 9 is: 5949
```

We show 25 samples of the dataset as an example below:

[3]:
```
fig, ax = plt.subplots(5, 5, figsize=(20,20))
for i in range(5):
    for j in range(5):
        ax[i][j].imshow(train_images[5 * i + j], cmap='Greys')
        ax[i][j].xaxis.set_visible(False)
        ax[i][j].yaxis.set_visible(False)
plt.show()
```



We reshape the data from (60000, 28, 28) to (60000, 784), to make it easier for calculations through the rest of project:

[4]:
```
train_images_reshaped = train_images.reshape((train_images.shape[0], 28*28))

print(f'Train images before reshape: {train_images.shape}')
print(f'Train images after  reshape: {train_images_reshaped.shape}')

assert all(train_images[0, 20] == train_images_reshaped[0, (20)*28:(20+1)*28]),␣
 ↪"train reshaping is wrong"
```

```
Train images before reshape: (60000, 28, 28)
Train images after  reshape: (60000, 784)
```

We will use the functions defined below to answer the questions:

[5]:
```python
def compute_pca_eig_vec(X, num_of_pca_components):
    covar_matrix = np.matmul(X.T , X)
    _, vectors = eigh(covar_matrix)
    descending_vectors = vectors[:,::-1]
    return descending_vectors[:, 0:num_of_pca_components]


def compute_x_reduced (X, eig_vectors):
    return np.matmul(X, eig_vectors)


def compute_x_reconstruct (X_reduced, eig_vectors):
    return np.matmul(X_reduced, eig_vectors.T)


def dispaly_eigs(eigs):
    fig, ax = plt.subplots(eigs.shape[0]//5, 5, figsize=(20, 20))
    for i in range(eigs.shape[0]//5):
        for j in range(5):
            ax[i][j].imshow(eigs[5 * i + j].reshape(28, 28), cmap='Greys')
            ax[i][j].xaxis.set_visible(False)
            ax[i][j].yaxis.set_visible(False)
    plt.show()


def dispaly_reconstruct(x_reconstruct):
    plt.imshow(x_reconstruct.reshape(28, 28), cmap='Greys')
    plt.axis('off')
    plt.show()


def different_pcas(X, num_of_eig):
    scal = StandardScaler()
    X_standard = scal.fit_transform(X)
    eig_vecs = compute_pca_eig_vec(X_standard, num_of_eig)
    print('The first 10 eigen-vectors:')
    dispaly_eigs(eig_vecs.T)

    X_reduced_2 = compute_x_reduced(X_standard[0:1], eig_vecs[:, 0:2])
    X_reduced_5 = compute_x_reduced(X_standard[0:1], eig_vecs[:, 0:5])
    X_reduced_10 = compute_x_reduced(X_standard[0:1], eig_vecs[:, 0:10])

    X_reconstruct_2 = compute_x_reconstruct(X_reduced_2, eig_vecs[:, 0:2])
    X_reconstruct_5 = compute_x_reconstruct(X_reduced_5, eig_vecs[:, 0:5])
    X_reconstruct_10 = compute_x_reconstruct(X_reduced_10, eig_vecs[:, 0:10])
```

```python
    print('Reconstruction of a sample from the first 2 eigen-vectors:')
    dispaly_reconstruct(scal.inverse_transform(X_reconstruct_2))
    print('Reconstruction of a sample from the first 5 eigen-vectors:')
    dispaly_reconstruct(scal.inverse_transform(X_reconstruct_5))
    print('Reconstruction of a sample from the first 10 eigen-vectors:')
    dispaly_reconstruct(scal.inverse_transform(X_reconstruct_10))

    return X_reconstruct_2, X_reconstruct_5, X_reconstruct_10


def different_pcas_Q2(X, num_of_eig):
    scal = StandardScaler()
    X_standard = scal.fit_transform(X)
    eig_vecs = compute_pca_eig_vec(X_standard, num_of_eig)
    print('The first 20 eigen-vectors:')
    dispaly_eigs(eig_vecs.T)

    digit_array = np.zeros((10, 784))
    for i in range(10):
        digit_array[i, :] = X_standard[train_labels == i, :][1]

    print('10 samples have been selected of every digit group:')
    dispaly_eigs(scal.inverse_transform(digit_array))

    X_reduced_2 = compute_x_reduced(digit_array, eig_vecs[:, 0:2])
    X_reduced_5 = compute_x_reduced(digit_array, eig_vecs[:, 0:5])
    X_reduced_10 = compute_x_reduced(digit_array, eig_vecs[:, 0:10])


    X_reconstruct_2 = compute_x_reconstruct(X_reduced_2, eig_vecs[:, 0:2])
    X_reconstruct_5 = compute_x_reconstruct(X_reduced_5, eig_vecs[:, 0:5])
    X_reconstruct_10 = compute_x_reconstruct(X_reduced_10, eig_vecs[:, 0:10])


    print('Reconstruction of samples from the first 2 eigen-vectors:')
    dispaly_eigs(scal.inverse_transform(X_reconstruct_2))
    print('Reconstruction of samples from the first 5 eigen-vectors:')
    dispaly_eigs(scal.inverse_transform(X_reconstruct_5))
    print('Reconstruction of samples from the first 10 eigen-vectors:')
    dispaly_eigs(scal.inverse_transform(X_reconstruct_10))

    return X_reconstruct_2, X_reconstruct_5, X_reconstruct_10
```

2 Question 1

In the first question we only need to know the first 10 eigen-vectors:

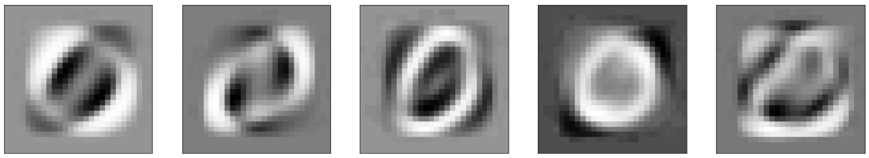
[6]:
```
num_of_eig = 10
```

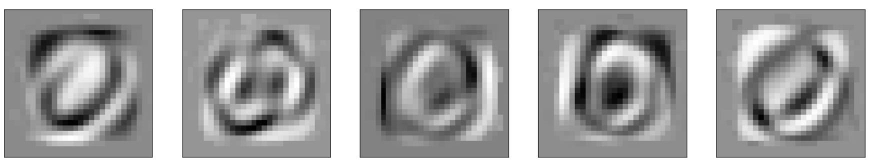
In the 10 following sectons, for group of unique digits, we will show the first 10 eigenvectors as images, and will reconstruct 3 versions of one sample of the group as listed below: - Reconstruction from the first 2 eigen-vectors - Reconstruction from the first 5 eigen-vectors - Reconstruction from the first 10 eigen-vectors

2.1 Digit 0

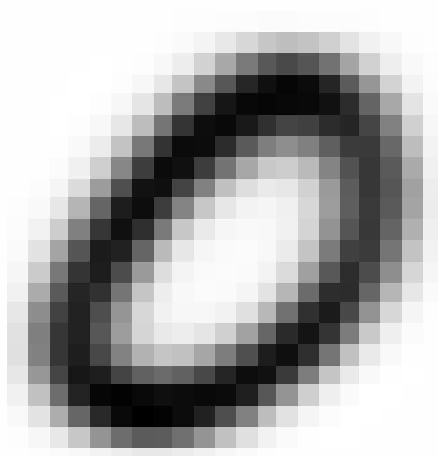
[7]:
```
X = train_images_reshaped[train_labels==0, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

The first 10 eigen-vectors:

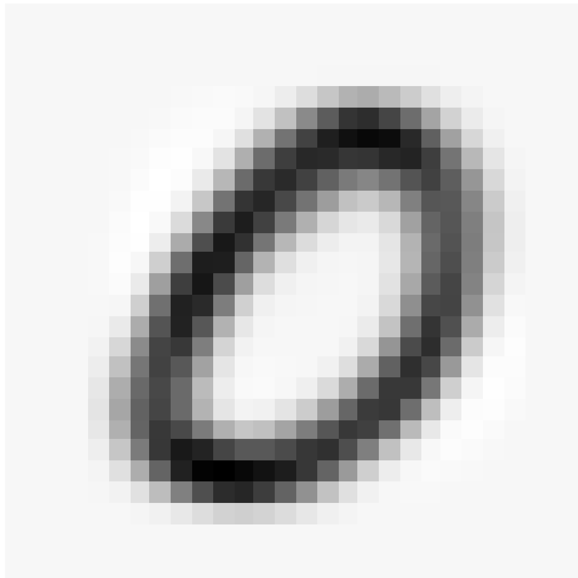




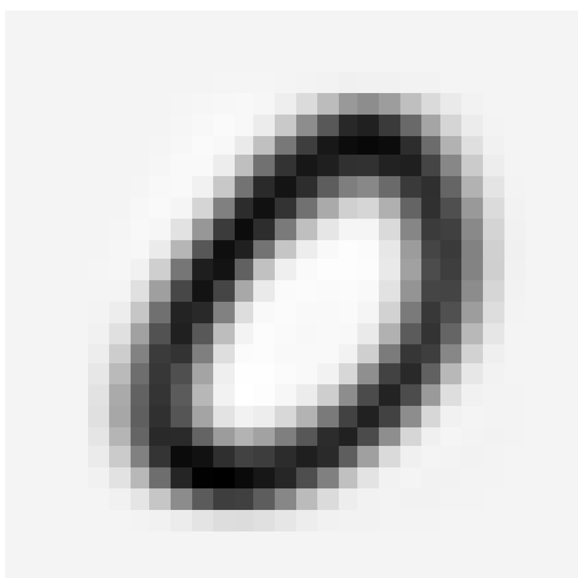
Reconstruction of a sample from the first 2 eigen-vectors:



Reconstruction of a sample from the first 5 eigen-vectors:



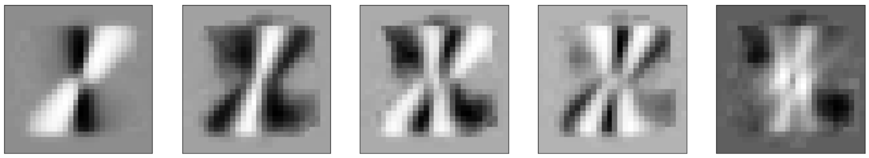
Reconstruction of a sample from the first 10 eigen-vectors:

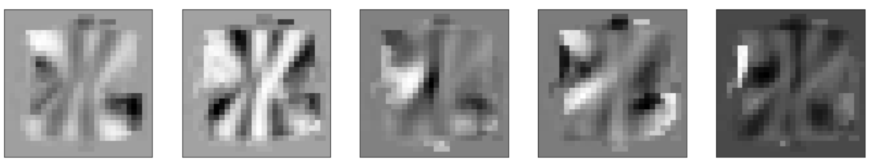


2.2 Digit 1

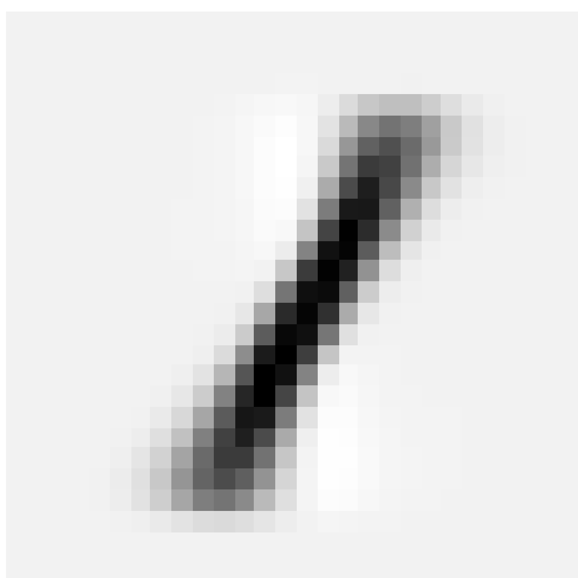
[8]:
```
X = train_images_reshaped[train_labels==1, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

The first 10 eigen-vectors:

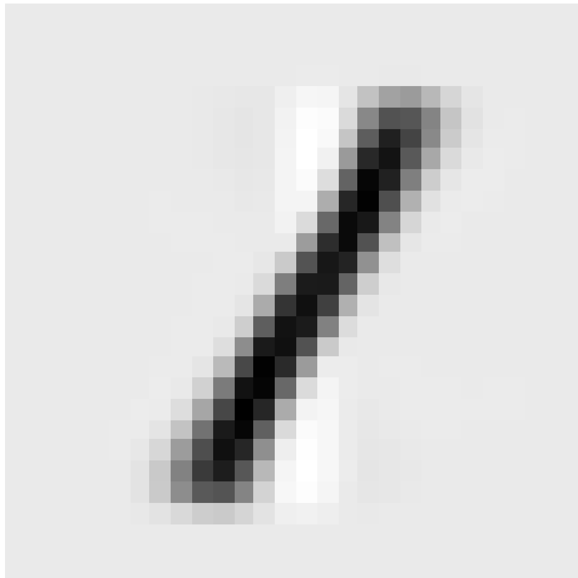




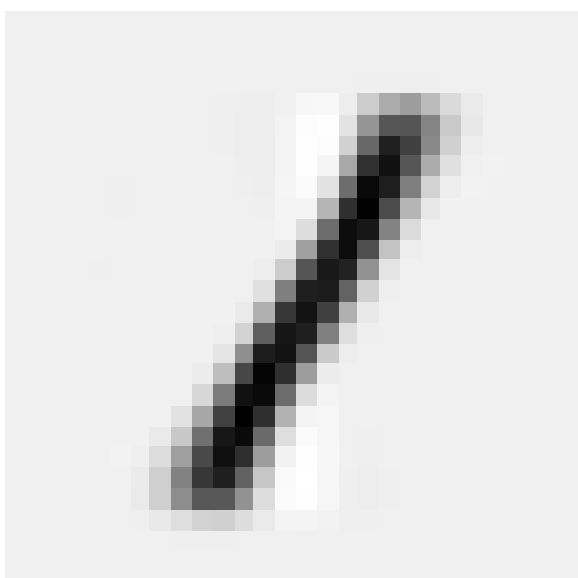
Reconstruction of a sample from the first 2 eigen-vectors:



Reconstruction of a sample from the first 5 eigen-vectors:



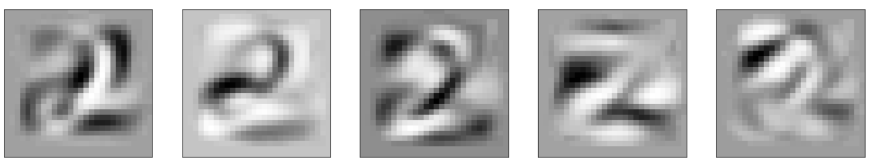
Reconstruction of a sample from the first 10 eigen-vectors:



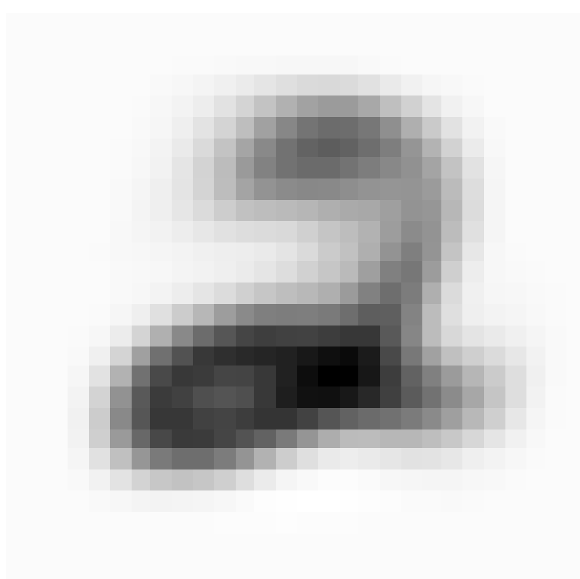
2.3 Digit 2

[9]:
```
X = train_images_reshaped[train_labels==2, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

The first 10 eigen-vectors:



Reconstruction of a sample from the first 2 eigen-vectors:



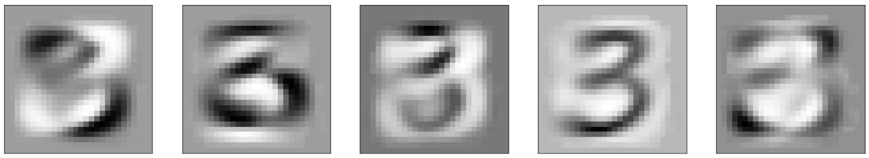
Reconstruction of a sample from the first 5 eigen-vectors:

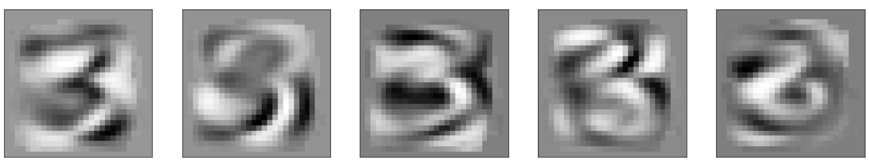
Reconstruction of a sample from the first 10 eigen-vectors:

2.4 Digit 3

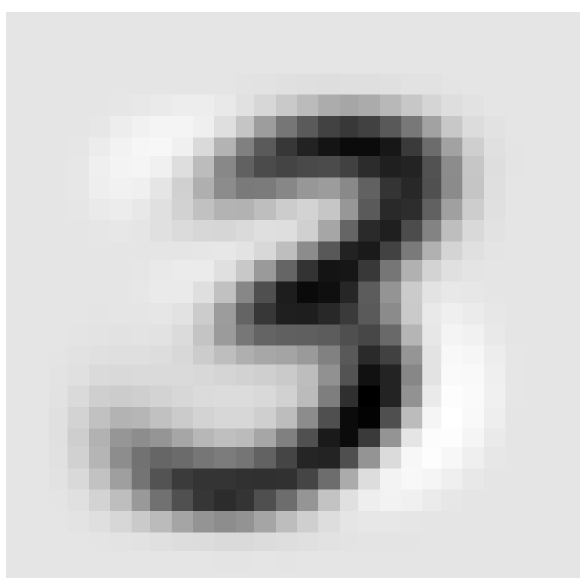
[10]:
```
X = train_images_reshaped[train_labels==3, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

The first 10 eigen-vectors:





Reconstruction of a sample from the first 2 eigen-vectors:



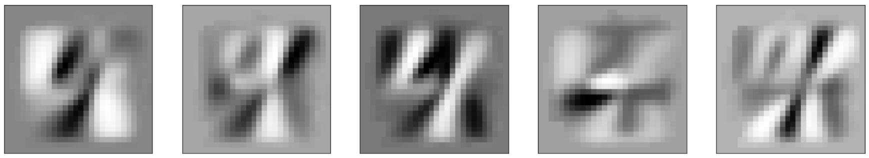
Reconstruction of a sample from the first 5 eigen-vectors:

Reconstruction of a sample from the first 10 eigen-vectors:

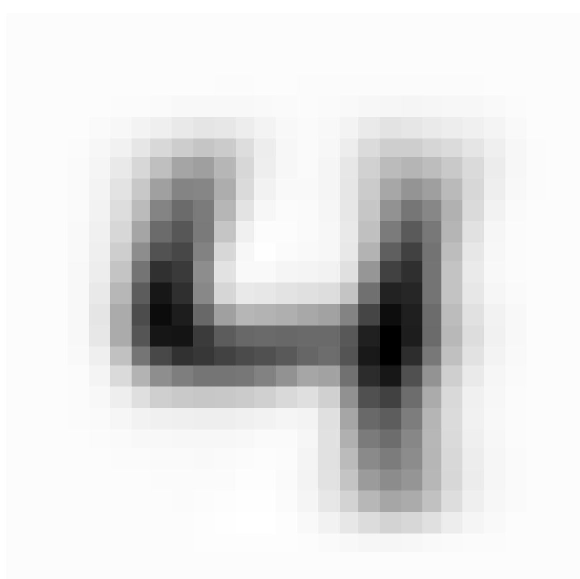
2.5 Digit 4

[11]:
```
X = train_images_reshaped[train_labels==4, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X,␣
↪num_of_eig)
```

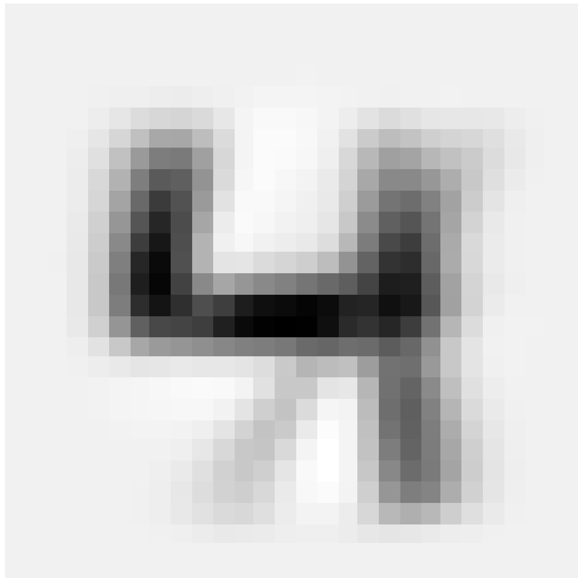
The first 10 eigen-vectors:



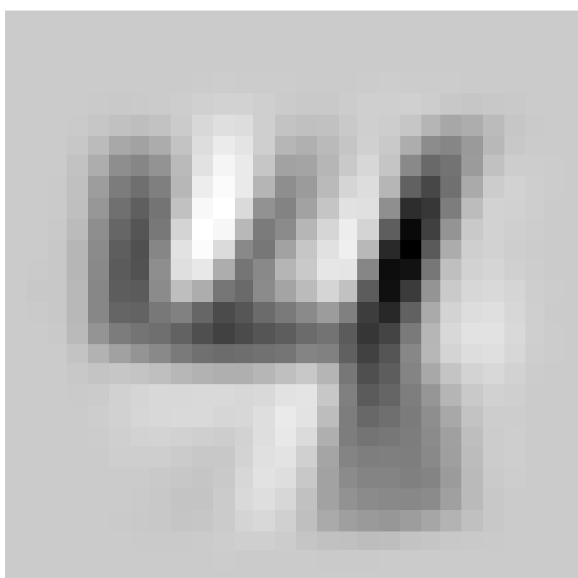
Reconstruction of a sample from the first 2 eigen-vectors:



Reconstruction of a sample from the first 5 eigen-vectors:



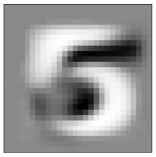
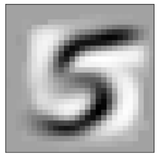
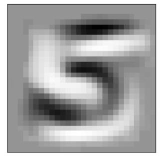
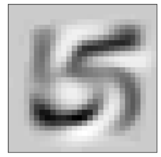
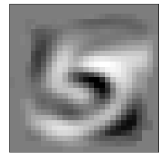
Reconstruction of a sample from the first 10 eigen-vectors:

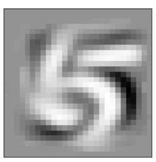
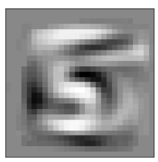
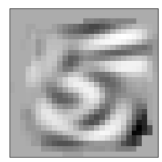


2.6 Digit 5

[12]:
```
X = train_images_reshaped[train_labels==5, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

The first 10 eigen-vectors:

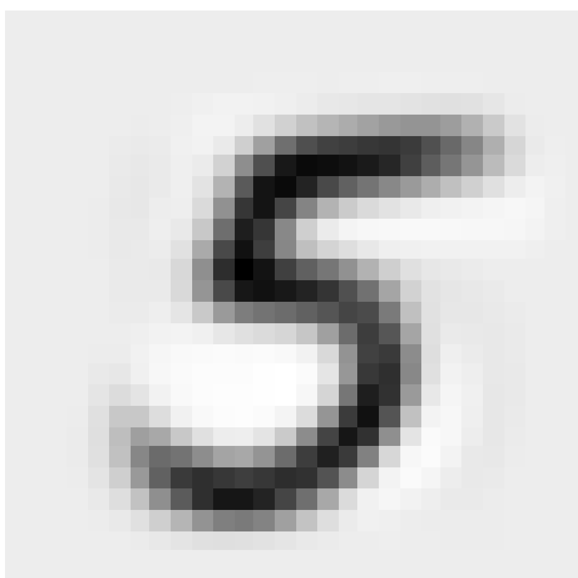
    

Reconstruction of a sample from the first 2 eigen-vectors:

Reconstruction of a sample from the first 5 eigen-vectors:

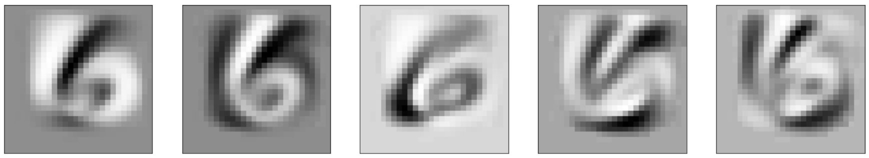
Reconstruction of a sample from the first 10 eigen-vectors:

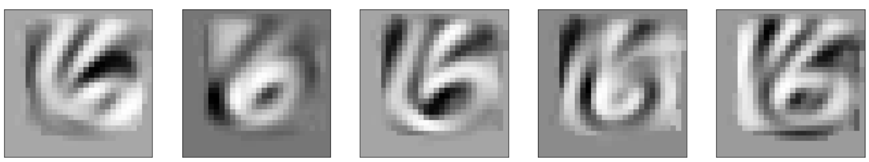


2.7 Digit 6

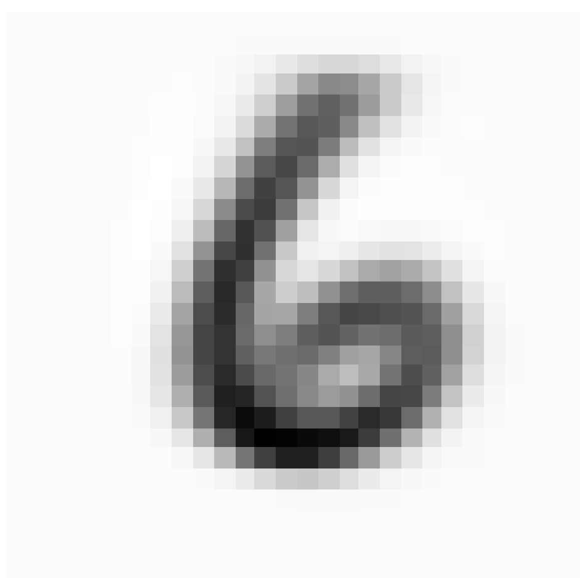
[13]:
```python
X = train_images_reshaped[train_labels==6, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

The first 10 eigen-vectors:



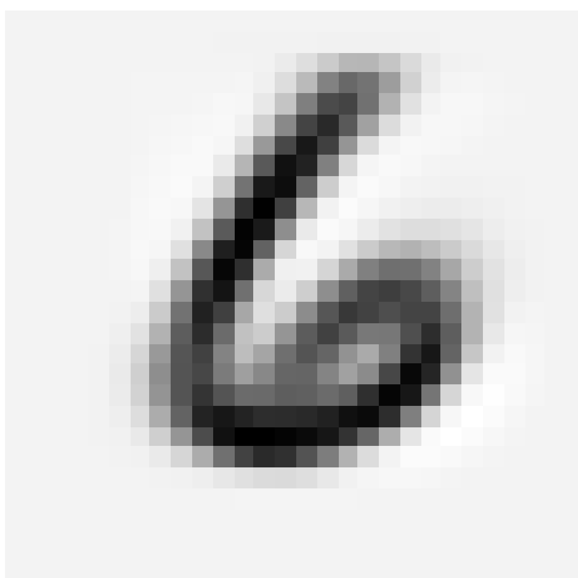


Reconstruction of a sample from the first 2 eigen-vectors:



Reconstruction of a sample from the first 5 eigen-vectors:

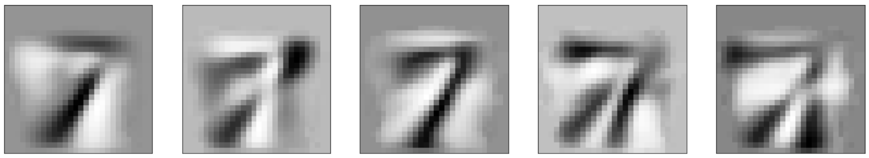
Reconstruction of a sample from the first 10 eigen-vectors:

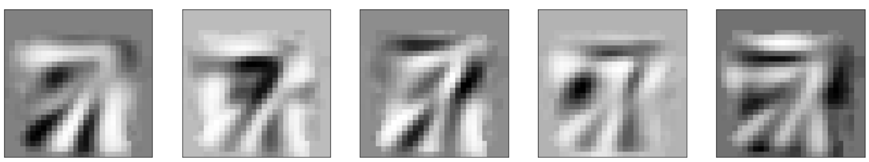


2.8 Digit 7

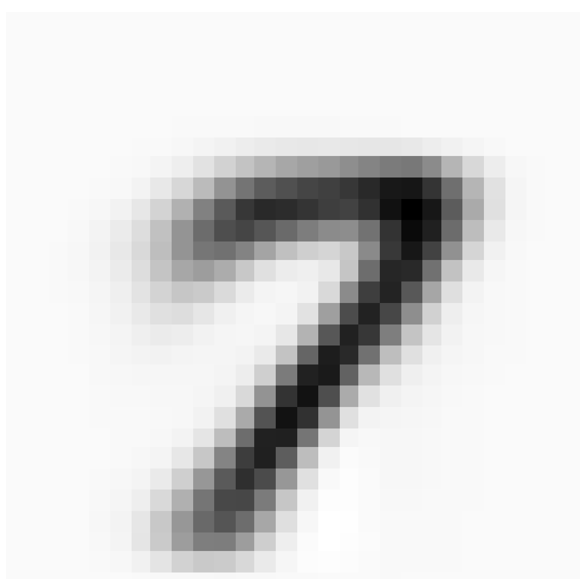
[14]:
```
X = train_images_reshaped[train_labels==7, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

The first 10 eigen-vectors:

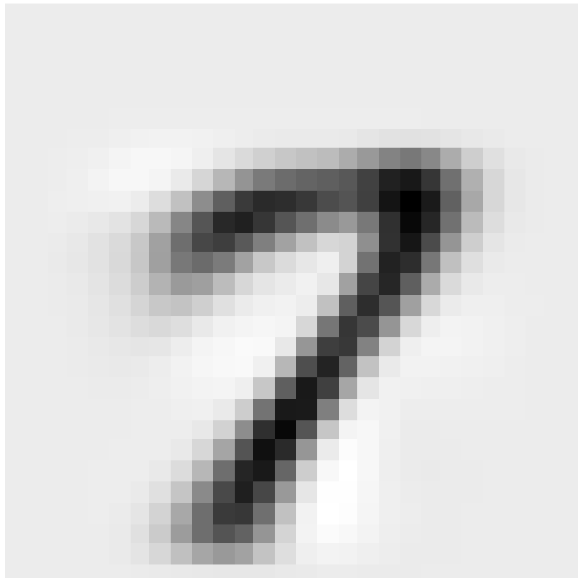




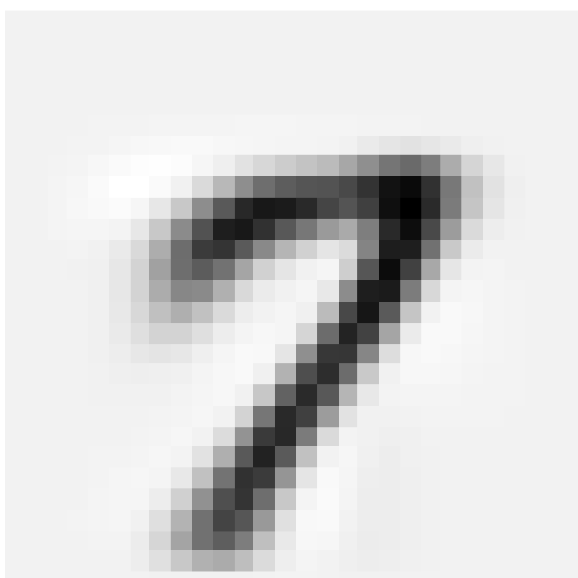
Reconstruction of a sample from the first 2 eigen-vectors:



Reconstruction of a sample from the first 5 eigen-vectors:



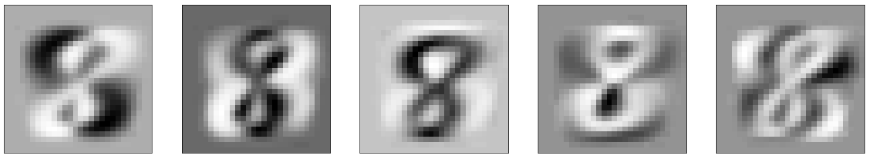
Reconstruction of a sample from the first 10 eigen-vectors:

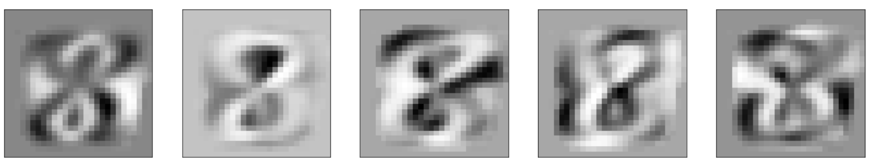


2.9 Digit 8

[15]:
```
X = train_images_reshaped[train_labels==8, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

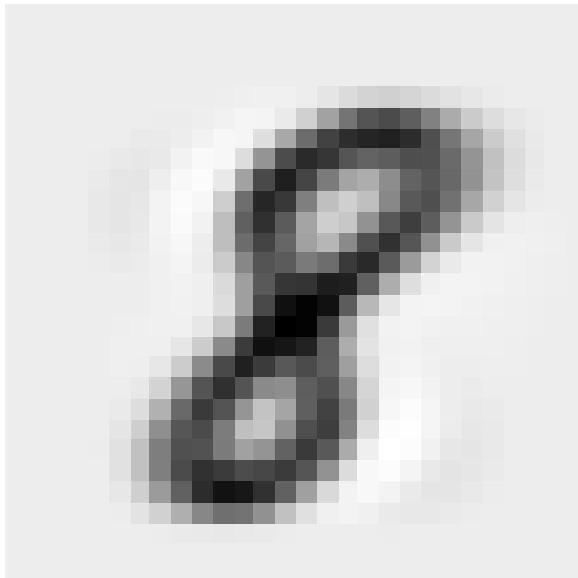
The first 10 eigen-vectors:



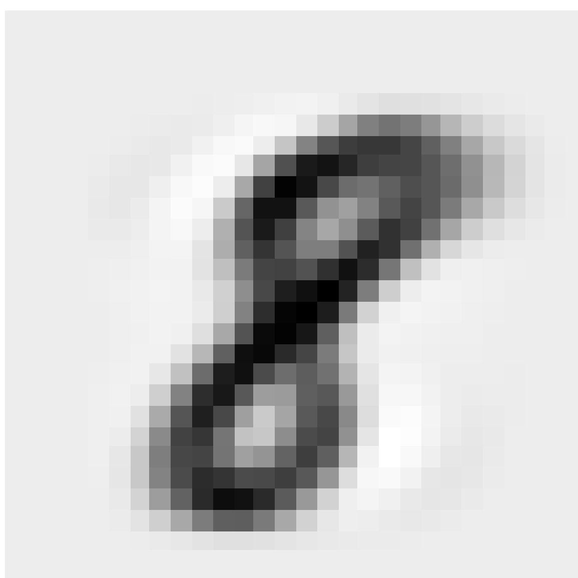


Reconstruction of a sample from the first 2 eigen-vectors:

Reconstruction of a sample from the first 5 eigen-vectors:



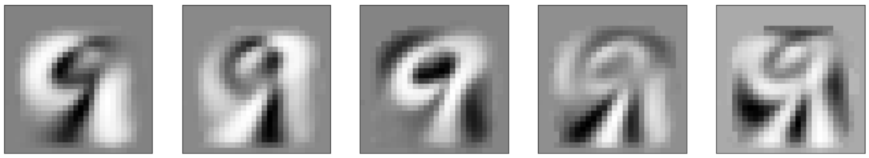
Reconstruction of a sample from the first 10 eigen-vectors:

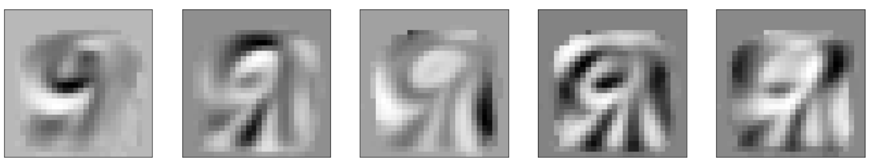


2.10 Digit 9

[16]:
```
X = train_images_reshaped[train_labels==9, :]
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas(X, num_of_eig)
```

The first 10 eigen-vectors:





Reconstruction of a sample from the first 2 eigen-vectors:

```
Reconstruction of a sample from the first 5 eigen-vectors:
```

```
Reconstruction of a sample from the first 10 eigen-vectors:
```

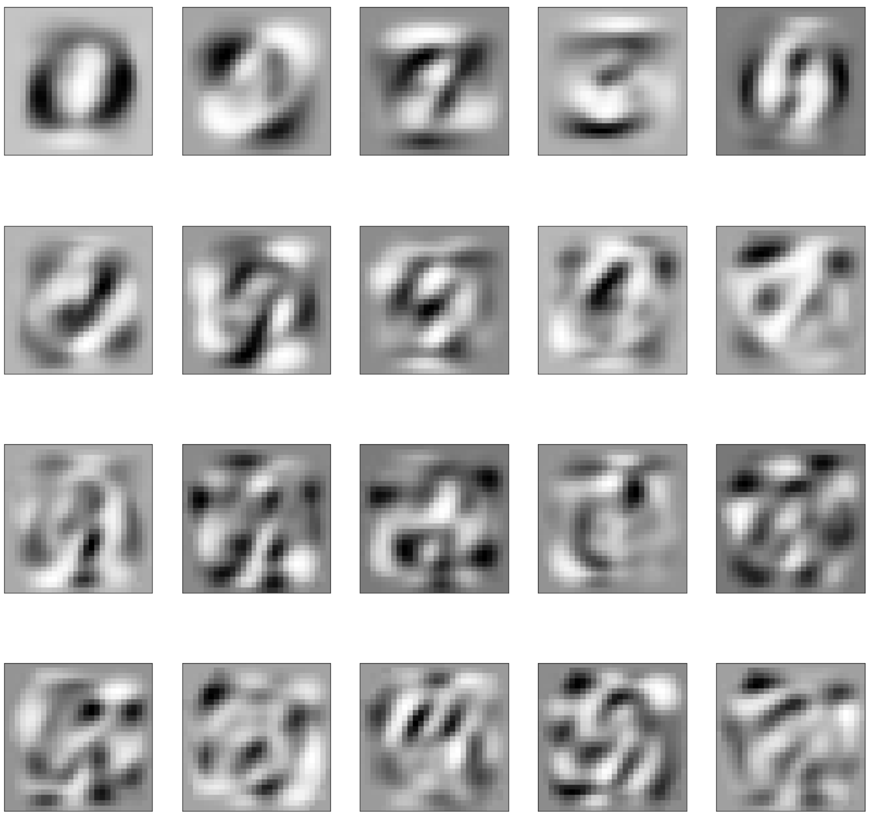
3 Question 2

In the second question, we only need to know the first 20 eigen-vectors:

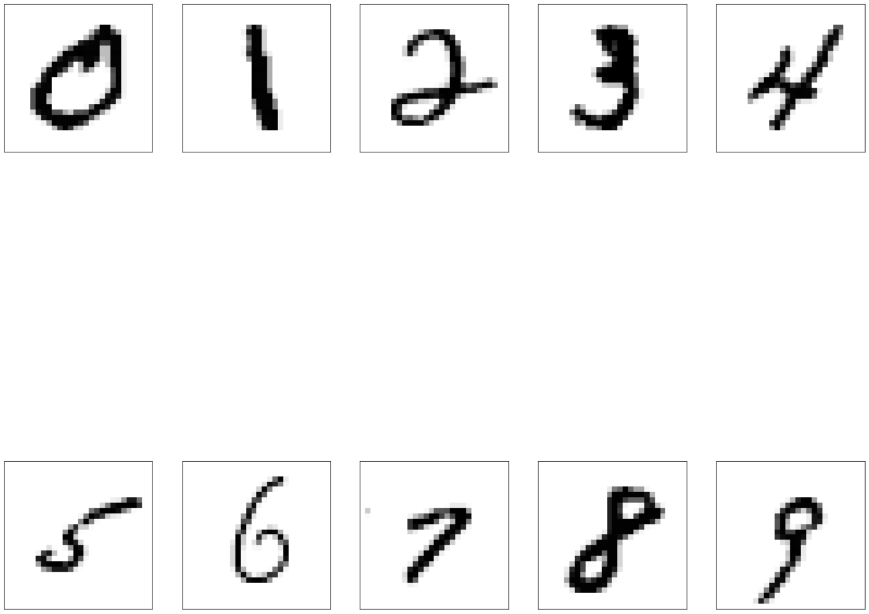
[17]:
```
num_of_eig = 20
```

[18]:
```
X = train_images_reshaped
X_reconstruct_2, X_reconstruct_5, X_reconstruct_10 = different_pcas_Q2(X,␣
↪num_of_eig)
```

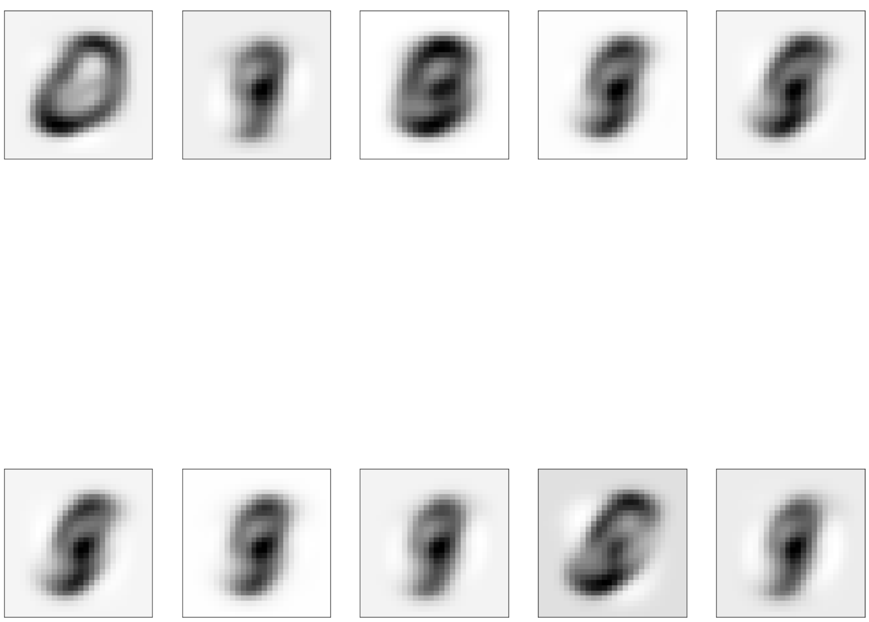
```
The first 20 eigen-vectors:
```



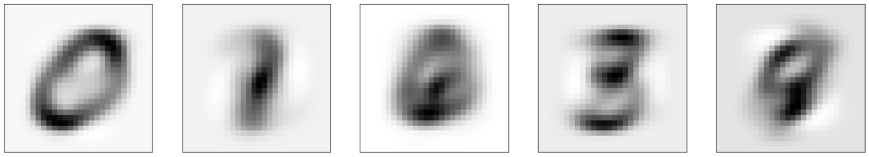
10 samples have been selected (one from every digit group), the original sampled images is shown below:

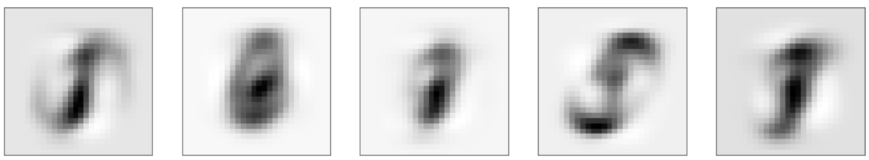


`Reconstruction of samples from the first 2 eigen-vectors:`



Reconstruction of samples from the first 5 eigen-vectors:





Reconstruction of samples from the first 10 eigen-vectors: