Lab Assignment - 7

CSL2010: Introduction To Machine Learning
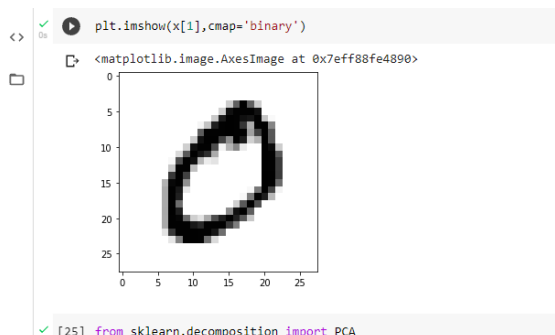
Principal Component Analysis (PCA)
AY 2021-22, Semester-I

TASK 1

1.1-Imported the dataset from sklearn.datasets.

1.2-The dataset contains the pixel values for images of numbers in form of 28*28 array(each image has 28*28 =784 pixel values.

1.3-

1.4-

(i)--for n=25 corresponding variance values=

Array-([0.09746112, 0.07155446, 0.06149538, 0.05403379, 0.04888936,
    0.04305226, 0.03278262, 0.02889639, 0.02758363, 0.02342141,
    0.02106691, 0.02037555, 0.01707064, 0.01694019, 0.01583382,
    0.01486345, 0.01319358, 0.01279013, 0.01187267, 0.01152933,
    0.01066029, 0.01009743, 0.00959038, 0.00909575, 0.00883034])

(ii)--for 80 percent variance the value of n=25

TASK 2

2.1-For making the covariance matrix first I took the mean of every column and stored it then subtracted each column mean from each data point in that column then multiplied this matrix with its transpose and divided it by 783(n=784).
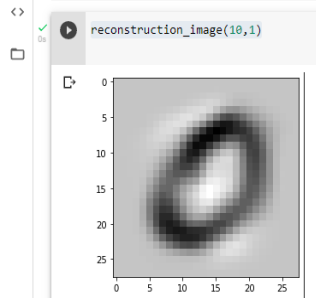
(I have used the starting 5000 values)

2.2-for obtaining the eigenvalues and eigenvectors i have used "eigen_values, eigen_vectors=np.linalg.eig(covariance_matrix)"

Then I sorted the eigenvalues and stored the sorted index.With help of these sorted index I sorted the eigenvectors accordingly.
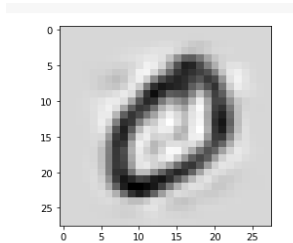
2.3-Reconstructed image using Eigenvector for each value of n

2.4-Reconstructed images:-For reconstructing the images I made a function called 'reconstruction_image' that takes number of components and the sample as its arguments.
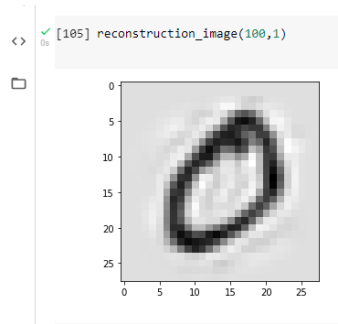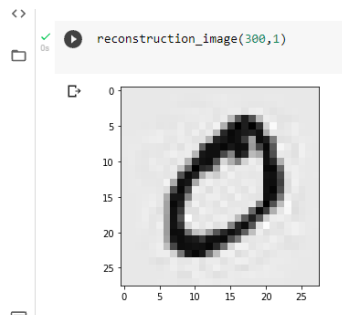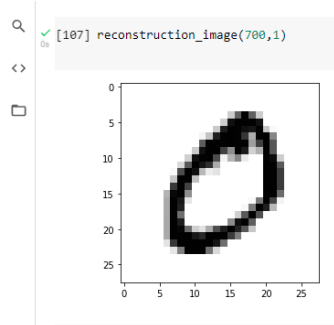
For 10 components

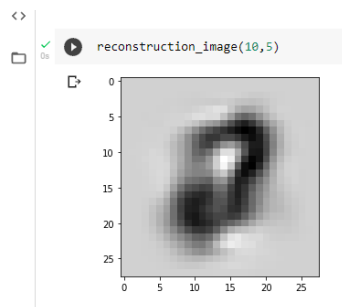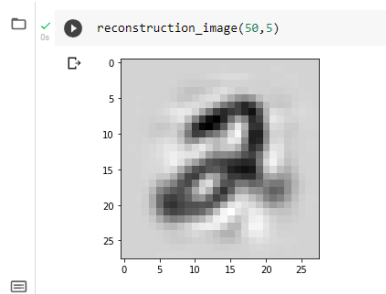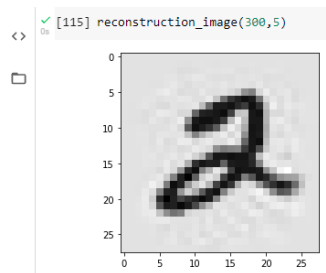For 50 components

For 100 components

For 300 components

```
reconstruction_image(300,1)
```


For 700 components

```
[107] reconstruction_image(700,1)
```


(ii)

For 10 components

```
reconstruction_image(10,5)
```


For 50 components

```
reconstruction_image(50,5)
```


For 100 components

reconstruction_image(100,5)

For 300 components

[115] reconstruction_image(300,5)

For 700 components

[118] reconstruction_image(700,5)
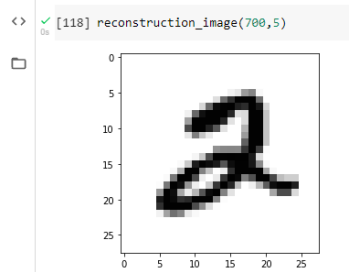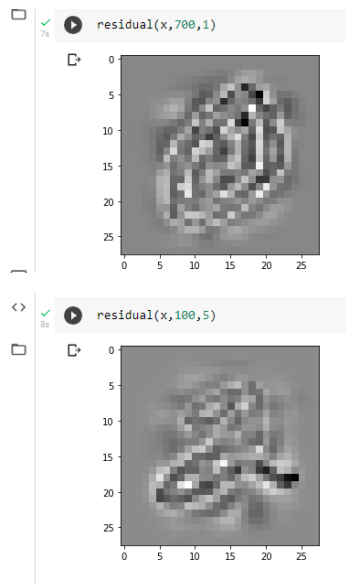
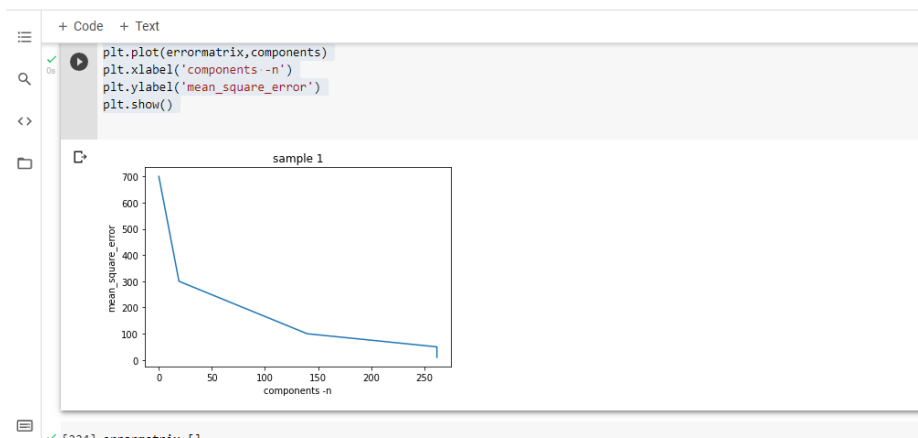As the number of components is increasing the quality of image also increases

2.5-drawn the residual images for each case by subtracting the pixel values in every case
For this i made a function 'residual' that takes x number of components and the sample as arguments.
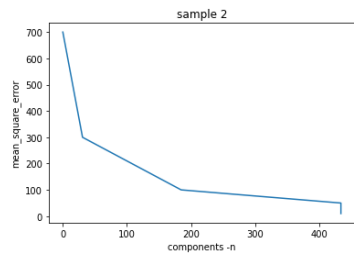Residual image for 0 and 2

```
residual(x,700,1)
```



```
residual(x,100,5)
```


2.6-for calculating the error i made a function 'error' that takes the number of components and the sample number as arguments and returns the error

Sample 1

```
plt.plot(errormatrix,components)
plt.xlabel('components -n')
plt.ylabel('mean_square_error')
plt.show()
```


Sample 2

sample 2

Copy_of_B20MT....ipynb   ⌃   B20MT006_Task....ipynb   ⌃