

Machine Learning Homework 2

Name: Mohamed Rissal Hedna
ID#: 811198495

Problem 1:(notebook)

Let: $y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$ $X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & x_4 \end{bmatrix}$

$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$

~~$y = XB + \epsilon$~~ $y = X\beta + \epsilon$

$e(\beta) = y - X\beta$

a) * Loss function:

$$L(\beta) = e^T e$$

$$= (y - X\beta)^T (y - X\beta)$$

$$= (y^T - \beta^T X^T)(y - X\beta)$$

$$= (y^T y - y^T X\beta - \beta^T X^T y + \beta^T X^T X\beta)$$

$$= y^T y - 2\beta^T X^T y + \beta^T X^T X\beta$$

b) Normal Equation:

$$X^T X \beta - X^T y = 0$$

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

The least square estimate.

~~show~~

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & x_4 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 1 & 5 \\ 1 & 3 \\ 1 & 10 \end{bmatrix}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 8 \\ 25 \\ 9 \\ 40 \end{bmatrix}$$

$X\beta$

$$X^T X \beta - X^T y = 0$$

$$0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 5 & 3 & 10 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 1 & 5 \\ 1 & 3 \\ 1 & 10 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 5 & 3 & 10 \end{bmatrix} \begin{bmatrix} 8 \\ 25 \\ 9 \\ 40 \end{bmatrix}$$

$$\begin{bmatrix} 4\beta_0 + 20\beta_1 - 82 \\ 20\beta_0 + 138\beta_1 - 568 \end{bmatrix} = 0 \Leftrightarrow \begin{bmatrix} 4 & 20 \\ 20 & 138 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} 82 \\ 568 \end{bmatrix}$$

c) The normal equation contains the factor X^T which projects the four-dimensional vectors into a two-dimensional plane allowing us to get the solution for the equation.

d) (Regression.py file)

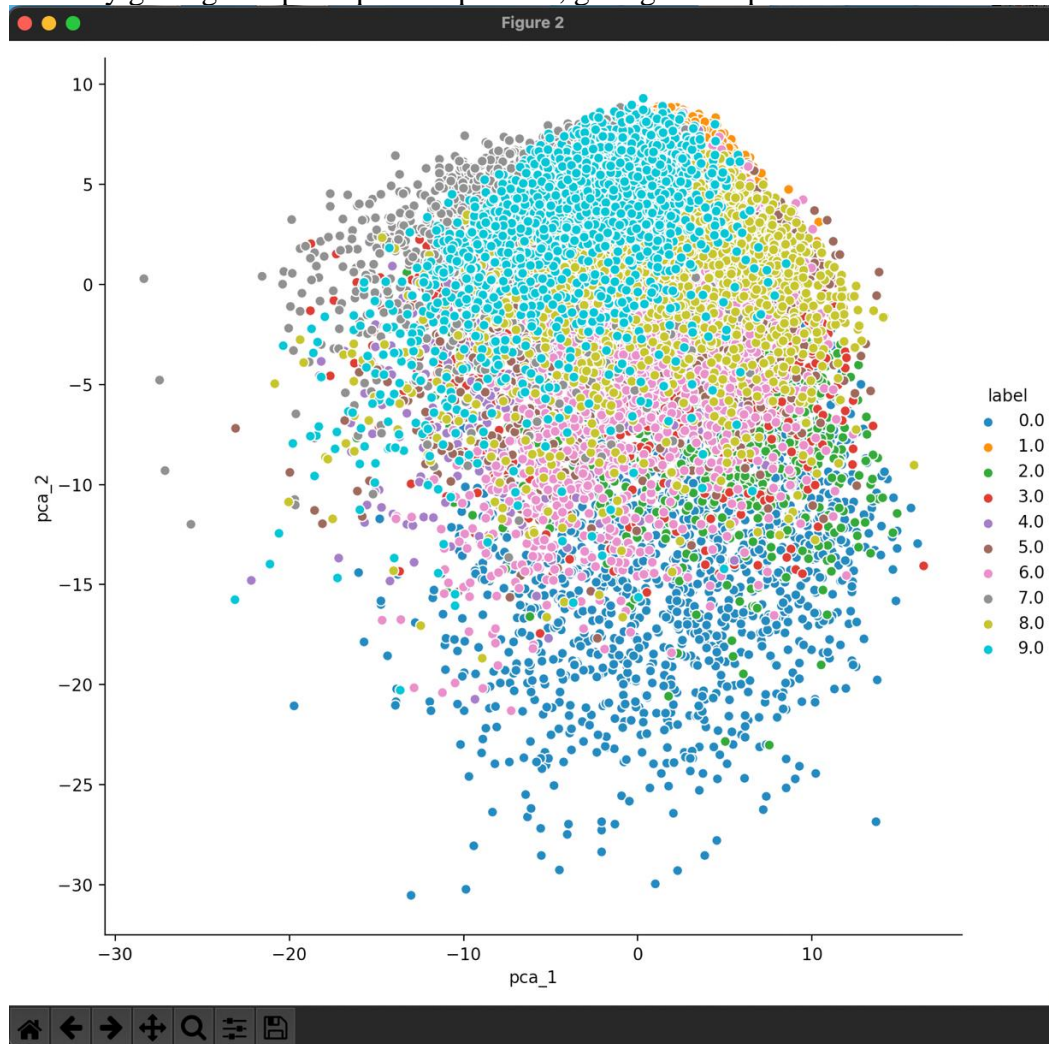
Problem 2 - PCA:

We performed PCA Analysis on the MNIST dataset

Files used:

- 1) PCA-MNIST.py (Python script)
- 2) train-labels.idx1-ubyte
- 3) t10k-labels.idx1-ubyte
- 4) t10k-images.idx3-ubyte
- 5) train-images.idx3-ubyte

After running the Python script on the above files, we reduce the dimensionality of the dataset to 2 dimensions by getting two principal components, giving us the plot below:



Python code: (Attached in zip file)











Problem 2 - PCA(a):

Files used:

- 1) avg-MNIST.py (Python script)
- 2) convert.py
- 3) test.csv
- 4) train.csv

For this exercise I had to convert the MNIST dataset into .csv files, the code for conversion is in “convert.py”. We can also find ready .csv files from the internet

Image means and standard deviations of each digit in the MNIST dataset after running the code in avg-MNIST.py on the files above:

Average Images										
Variance	0.3477	0.2443	0.3259	0.3179	0.2975	0.3036	0.3149	0.2917	0.3253	0.2986

Python code: (Attached in zip file)

Problem 2 - PCA(b): (notebook)

Let X be the data matrix of size $n \times b$,
then, the principal components are coming
from the eigenvectors of the covariance
matrix:

$$C = \frac{1}{n-1} X^T X.$$

Given that the SVD of X is given as:

$$X = U \Sigma V^T \text{ then:}$$

$$C = \frac{1}{n-1} X^T X = \frac{1}{n-1} (U \Sigma V^T)^T \cdot U \Sigma V^T$$

$$= \frac{1}{n-1} V \Sigma U^T \cdot U \Sigma V^T$$

$$= \frac{1}{n-1} V \Sigma^2 V^T$$

Thus, the columns of V are the eigenvectors
C. Therefore if the SVD of X is

$$X = U \Sigma V^T$$

then the columns of V give the
principal components:

$$XV = U \Sigma V^T \cdot V = \boxed{U \Sigma}$$

The formula for the eigenvectors of X
~~it~~ would be derived from the diagonal
elements of the matrix Σ since :

$$\lambda_{xi} = \sigma_{xi}^2$$

ith eigenvalue
of X

ith diagonal element
of Σ

Problem 3 - PCA(c):

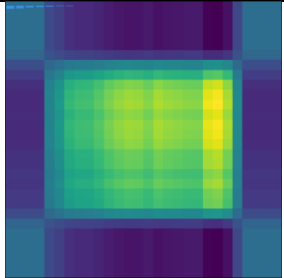
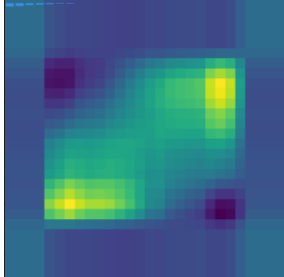
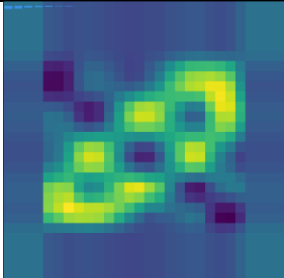
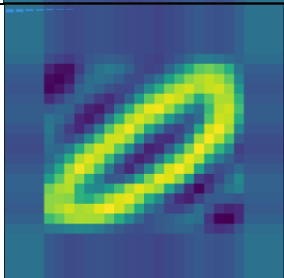
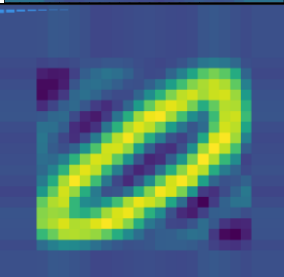
Files used:

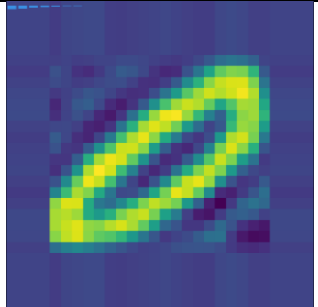
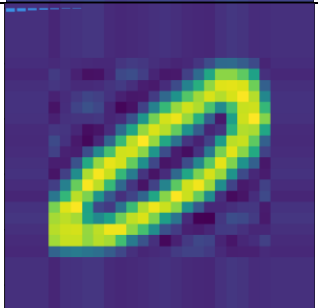
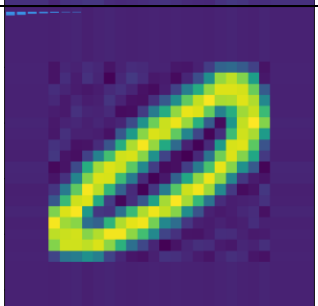
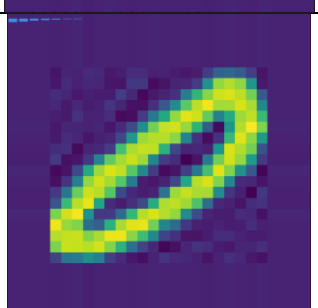
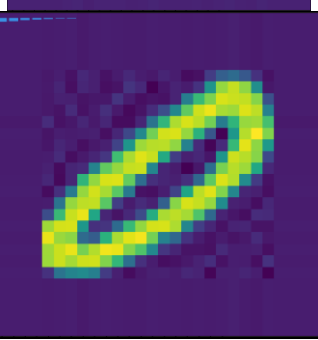
- 1) reconstruct.py (Python script)

The reconstruct.py code has two parts, the first part is a code snippet that chooses a random matrix from the MNIST dataset, converts it into an Image usable by python, and stores it in the variable "data_grid".

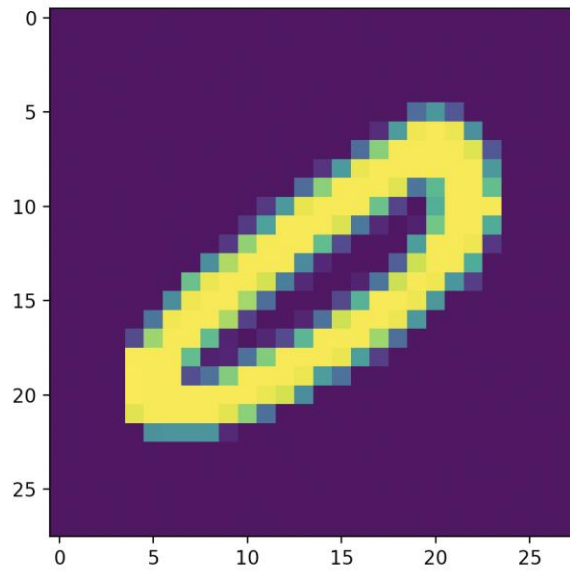
The second part is the solution to this part of the problem which is a function that takes a matrix and an integer value as its arguments, returns the top k principal components, and displays the image before/after dimensionality reduction. We also display a graph of how much variance is explained by each eigenvector.

After running this code on the digit zero for the first 10 principal components, we get the following results:

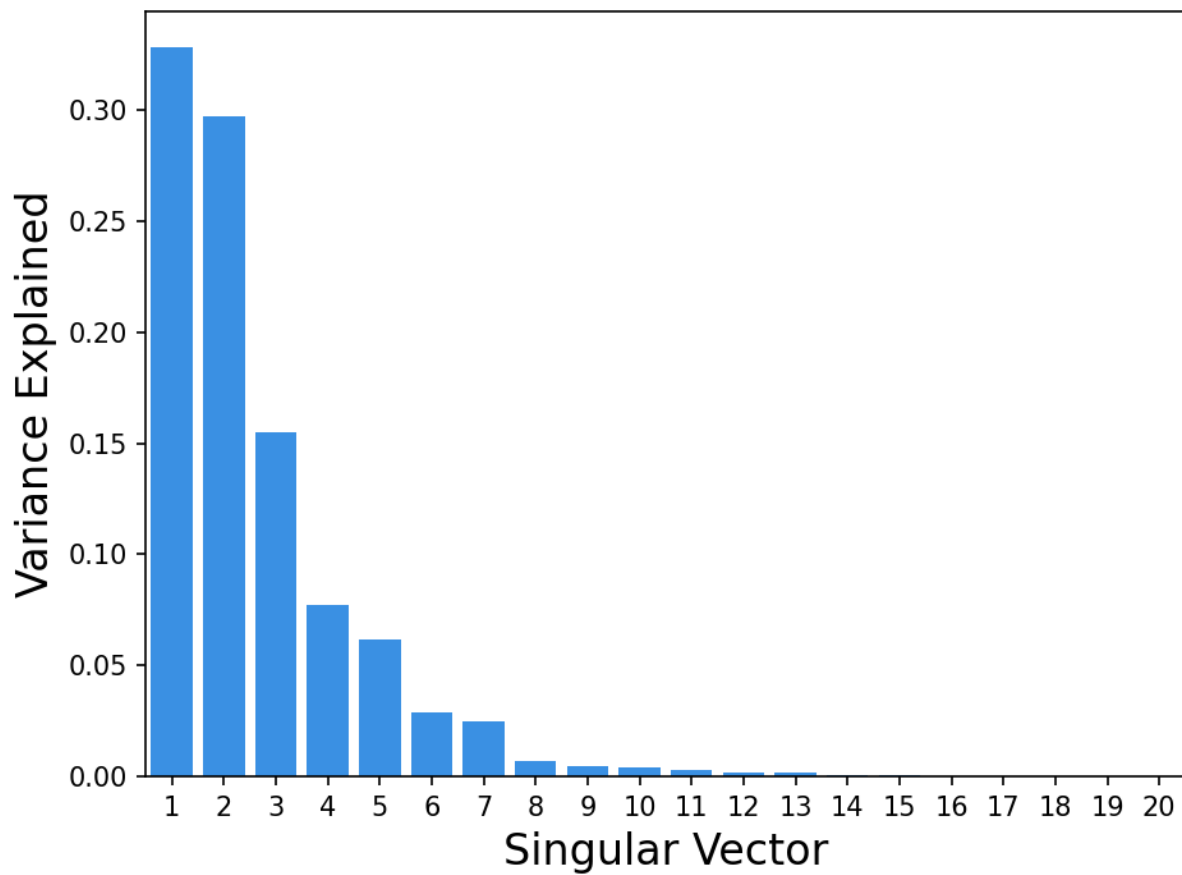
Number of components	Reconstructed images	
1		
2		
3		
4		
5		

6	
7	
8	
9	
10	

The original image before reconstruction:



The variance explained by each eigenvector sorted from greatest to smallest is visualized in the graph below:

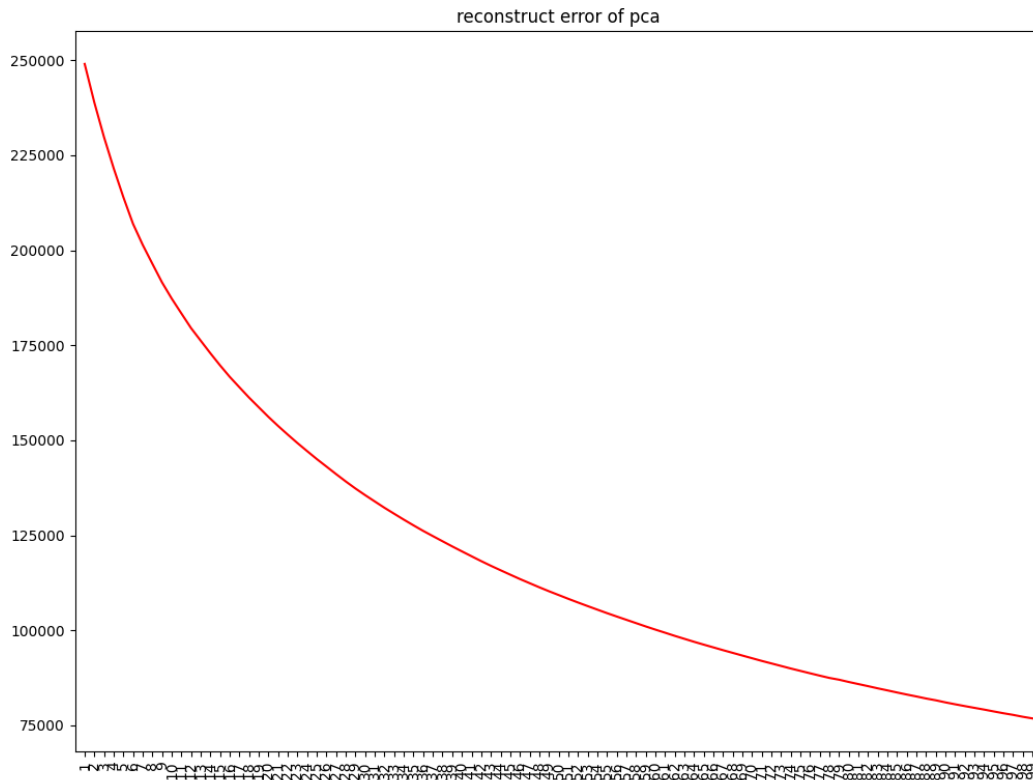


Problem 3 – PCA(d):

Files used:

1) reconstruct.py

After running the script for 784 eigenvectors, my laptop was still running after an hour and was heating up which is why I lowered the number to 100 eigenvectors instead, and visualized it in the following graph:



We can see that the more components and dimensions we add, the less the reconstruction error.

Maximum Likelihood Estimation for Multivariate Gaussians: (notebook)

a) Multivariate Gaussian Distribution:

$$\mathcal{N}(x; \mu, \Sigma) =$$

$$\frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} (x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

b) Defining the MG based on linear maps:

$$x \rightsquigarrow \mathcal{N}(\mu, \Sigma)$$

$$y \rightsquigarrow \mathcal{N}(A\mu + b, A\Sigma A^T)$$

$$y = Ax + b$$

c) Explicit form of the log-likelihood:

Univariate Gaussian: $\mathcal{N}(x; \mu, \sigma^2) =$

$$\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}\right)$$

$$\cancel{L(\mu, \sigma^2)} \quad L(\mu, \sigma^2 | x_1, \dots, x_n) = L(\mu, \sigma^2 | x_1)$$

$$L(\mu, \sigma^2 | x_2)$$

$$L(\mu, \sigma^2 | x_n)$$

$$= \frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(x_1-\mu)^2}{2\sigma^2}} \times \dots \times \frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(x_n-\mu)^2}{2\sigma^2}}$$

$$\begin{aligned} \ln(L(\mu, \sigma | x_1, \dots, x_n)) &= \ln\left(\frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(x_1-\mu)^2}{2\sigma^2}} \times \dots\right) \\ &= \underbrace{\ln\left(\frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(x_1-\mu)^2}{2\sigma^2}}\right)}_{n \text{ times}} + \ln(\dots) + \dots + \ln(\dots) \end{aligned}$$

$$\Rightarrow \text{we take } \ln\left(\frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{(x_n-\mu)^2}{2\sigma^2}}\right)$$

$$\Downarrow$$

$$\ln(L(\mu, \sigma | x_n)) = -\frac{1}{2} \ln(2\pi) - \ln(\sigma^2) - \frac{(x_n-\mu)^2}{2\sigma^2}$$

Therefore:

$$\ln(L(\mu, \sigma | x_1, \dots, x_n)) = -\frac{n}{2} \ln(2\pi) - n \ln(\sigma^2) - \frac{(x_1-\mu)^2}{2\sigma^2} - \dots - \frac{(x_n-\mu)^2}{2\sigma^2}$$

d) The closed form solution based on MLE for the mean:

- We take the derivative with respect to μ

$$\frac{d}{d\mu} (L(\mu, \sigma | x_1, \dots, x_n)) =$$

$$] \quad 0 - 0 - \frac{(x_1 - \mu)}{\sigma^2} + \dots + \frac{(x_n - \mu)}{\sigma^2}$$

$$] = \frac{1}{\sigma^2} [(x_1 + \dots + x_n) - n\mu]$$

We put the derivative equal to zero:

$$\frac{d(L(\mu, \sigma | x_1, \dots, x_n))}{d\mu} = 0$$

$$\frac{1}{\sigma^2} [(x_1 + \dots + x_n) - n\mu] = 0$$

$$\mu = \frac{x_1 + \dots + x_n}{n}$$

$$\boxed{\mu = \frac{\sum_{i=0}^n x_i}{n}}$$

The derivative with respect to σ :

$$\begin{aligned}\frac{d(L(\mu, \sigma | x_1, \dots, x_n))}{d\sigma} &= \sigma - \frac{n}{\sigma} + \frac{(x_1 - \mu)^2}{\sigma^3} + \dots + \frac{(x_n - \mu)^2}{\sigma^3} \\ &= -\frac{n}{\sigma} + \frac{1}{\sigma^3} \left[(x_1 - \mu)^2 + \dots + (x_n - \mu)^2 \right]\end{aligned}$$

We put it equal to zero:

$$\frac{1}{\sigma^2} \left[(x_1 - \mu)^2 + \dots + (x_n - \mu)^2 \right] - \frac{n}{\sigma} + \frac{1}{\sigma^3} \left[(x_1 - \mu)^2 + \dots + (x_n - \mu)^2 \right] = 0$$

$$\Rightarrow \sigma = \sqrt{\frac{(x_1 - \mu)^2 + \dots + (x_n - \mu)^2}{n}}$$