PYTHON機器學習入門

UNIT 8：DIMENSIONAL REDUCTION

授課教師：江尚瑀

THE IDEAL SET IF FEATURES

- High Variance: features with a lot of variance contain a lot of potential signal — signal (a.k.a. useful information) is a basic requirement for building a good model.
- Uncorrelated: features that are highly correlated with each other are less useful and in certain cases downright harmful (when the correlation is so high as to cause multicollinearity).
- Not That Many: a low number of features relative to our number of target variable observations.

FEATURE SELECTION

- **Removing features with low variance**

$$Var(X) = E[(X - \mu)^2]$$

- **Univariate feature selection**

  單獨計算每個特徵的統計值來決定最重要的**K**個特徵**(SelectKBest)**

  或排名前多少百分比的特徵**(SelectPercentile)**

  8.1_Feature Selection.ipynb

DIMENSIONALITY REDUCTION - PCA

INTRODUCTION TO PCA

- **PCA (Principal Component Analysis)**
  - **An effective method for reducing a dataset's dimensionality while keeping spatial characteristics as much as possible**
  - 用**m**維來代表**n**維空間**(m<n)**，用少數的維度來代表全部的維度，並找到新的座標定義方式，使得每一點投影在新的座標上時只用到少量的資訊和維度。

Characteristics:

- For unlabeled data
- A linear transform with solid mathematical foundation

Applications

Line/plane fitting

Face recognition

Machine learning

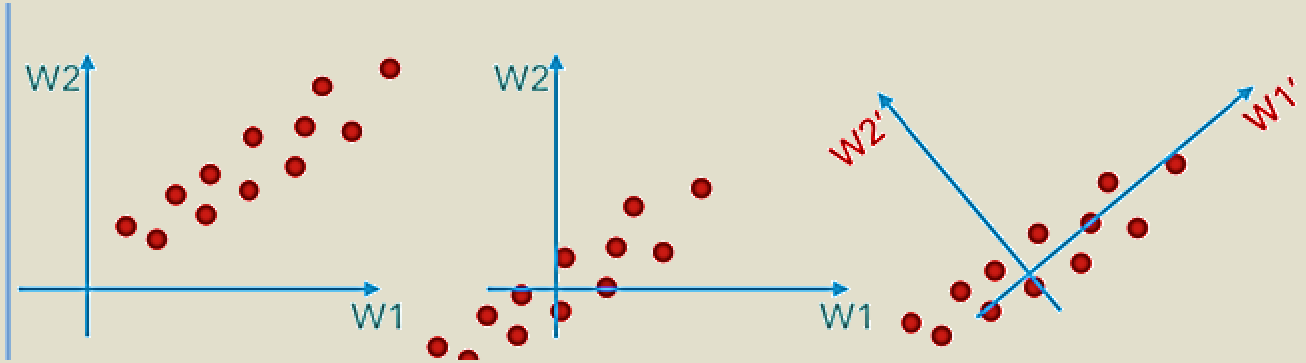
…

INTRODUCTION TO PCA

- **PCA (Principal Component Analysis)**
  - 原始數據拆解成更具代表性的主成分，並以其作為新的基準，重新描述數據
  - 目的是希望資料經過轉換過後可以保留最大的變異

$$\mathrm{x} = [x_1, x_2, \cdots, x_d] \text{，} x \in R^d$$

$$\downarrow \mathrm{xW} \text{，} \mathrm{W} \in R^{dxk}$$



$$\mathrm{z} = [z_1, z_2, \cdots, z_d] \text{，} z \in R^k$$

$$\Sigma_Z = E(\mathbf{z}\mathbf{z}^T) = AE(\mathbf{x}\mathbf{x}^T)A^T = A\Sigma_X A^T = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_p^2 \end{bmatrix}$$

PCA 演算法步驟(1/2)

假設原始數據有 **d** 維特徵

1. 標準化數據，將 **bias** 調成 **0, stdev** 調成**1**
2. 建立共變異數矩陣**(covariance matrix)**
3. 分解共變異數矩陣**(covariance matrix)**
   → **A** 特徵向量**(eigenvector)**與 **λ** 特徵值**(eigenvalues)**

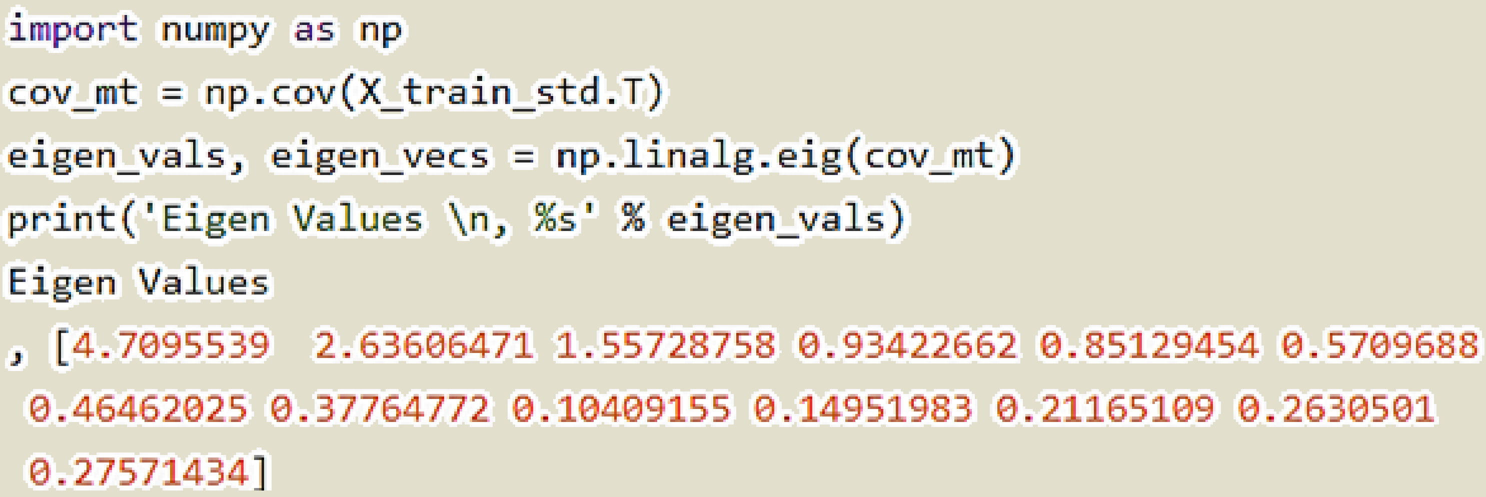
$$\sigma_{jk} = \frac{1}{n}\Sigma_{i=1}^{n}(x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k)$$

$$A = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 \end{bmatrix}$$

$$Av = \lambda v$$

PCA 演算法步驟(2/2)

$$Av = \lambda v$$



```
import numpy as np
cov_mt = np.cov(X_train_std.T)
eigen_vals, eigen_vecs = np.linalg.eig(cov_mt)
print('Eigen Values \n, %s' % eigen_vals)
Eigen Values
, [4.7095539  2.63606471 1.55728758 0.93422662 0.85129454 0.5709688
 0.46462025 0.37764772 0.10409155 0.14951983 0.21165109 0.2630501
 0.27571434]
```

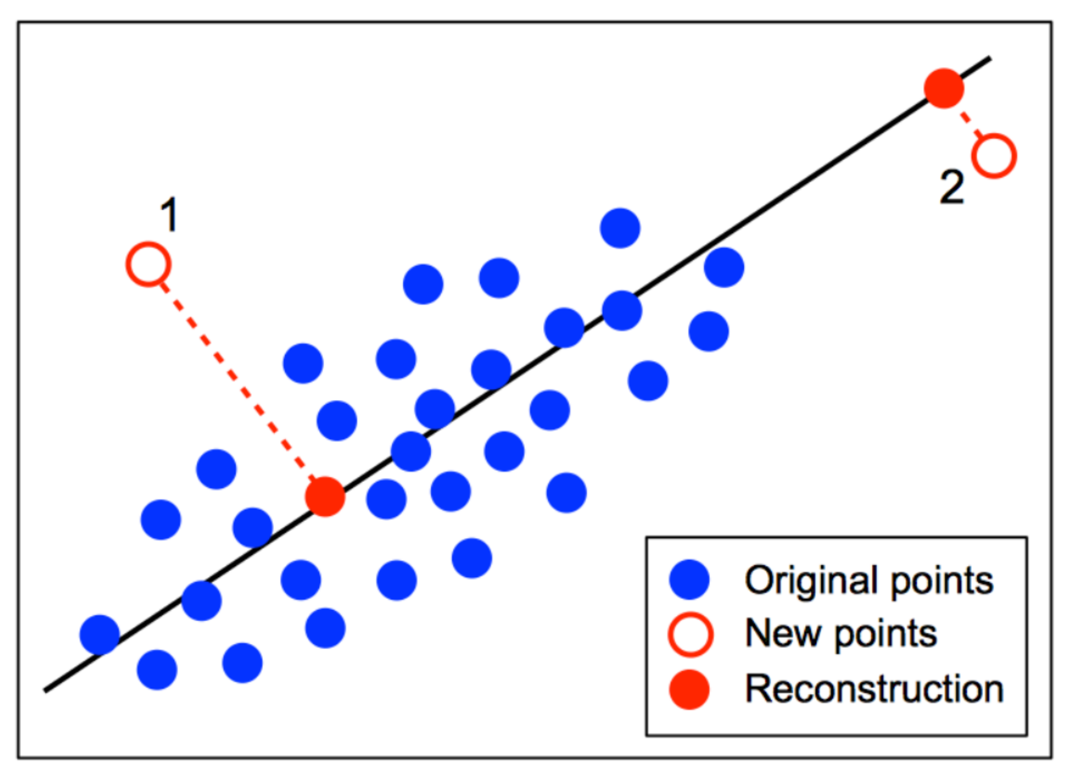
4. 選取 **k** 個最大特徵值相對應的特徵向量，其中 **k** 即為新特徵子空間的維數
5. 使用排序最上面的 **k** 個的特徵向量，建立投影矩陣**(project matrix)W**
6. 使用投影矩陣**(project matrix) W** 轉換原本 **d** 維的原數據至新的 **k** 維特徵子空間

PCA 異常值偵測

- PCA 是由一個原本矩陣 **X** 乘上投影矩陣 **W** 得到的結果
- 從 PCA 的結果乘上 **W** 的轉置矩陣來還原到原本的矩陣空間

$$\underset{n\times k}{Z} = \underset{n\times d}{X} \times \underset{d\times k}{W}$$

$$\underset{n\times d}{\bar{X}} = \underset{n\times k}{Z} \times \underset{k\times d}{W}^{\top}$$



PCA 優缺點

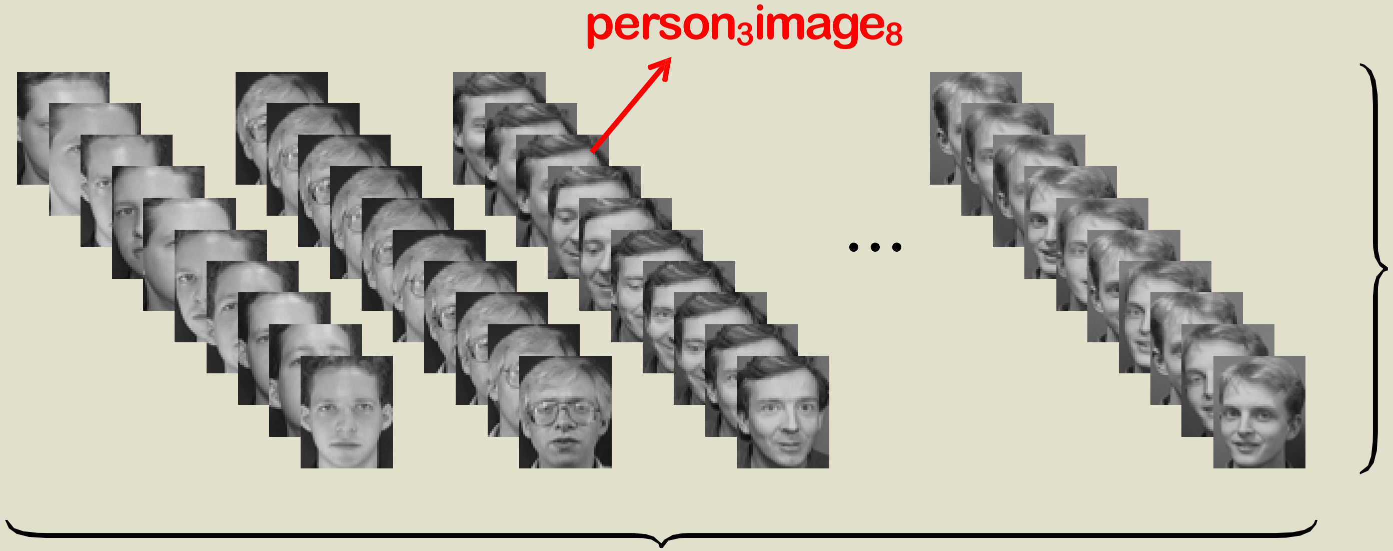
- 優點
  - 僅以方差衡量資訊量，不受資料集以外的因素影響
  - 各主成分之間正交，可消除原始資料各變數之間的相互影響
  - 方法簡單，易於實現
- 缺點
  - 各個主成分的含義具有模糊，解釋性弱，通常只有資訊量而無實際含義
  - 在樣本非正態分佈時得到的主成分不是最優的，因此特殊情況下方差小的成分也可能含有重要資訊

- **PCA**延伸閱讀**https://leemeng.tw/essence-of-principal-component-analysis.html**

APPENDIX

LAB – FACE RECOGNITION

- **Perform PCA as taught in the lectures**
- **Dataset**



person$_3$image$_8$

…

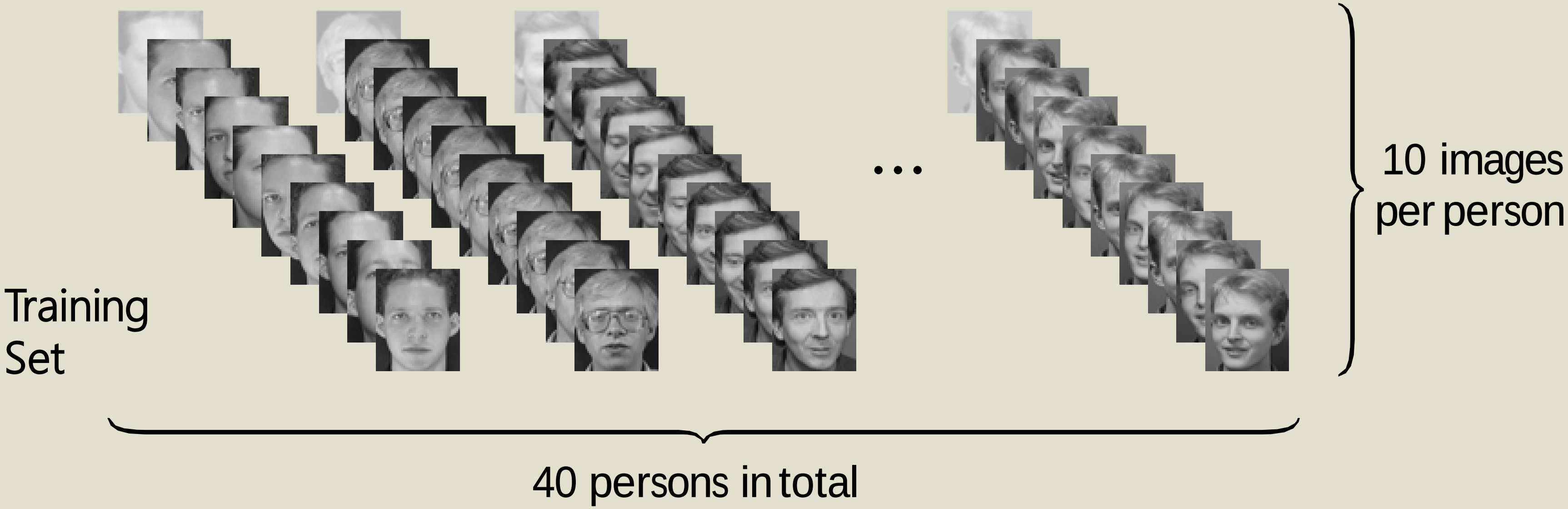
10 images per person

40 persons in total

The lab practice referred to NTU Deep Learning for Computer Vision (DLCV) course website.

LAB – FACE RECOGNITION

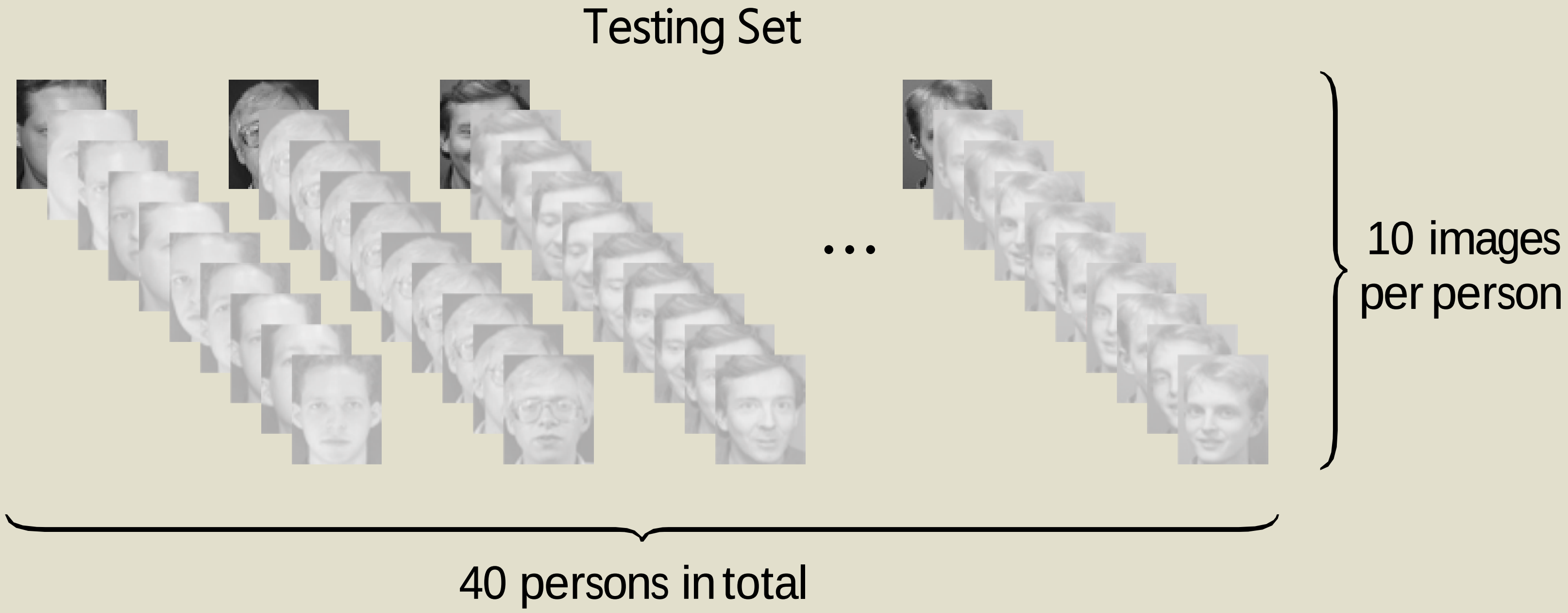
- **Perform PCA as taught in the lectures**
- **Dataset**



The lab practice referred to NTU Deep Learning for Computer Vision (DLCV) course website.

LAB – FACE RECOGNITION

- **Perform PCA as taught in the lectures**
- **Dataset**

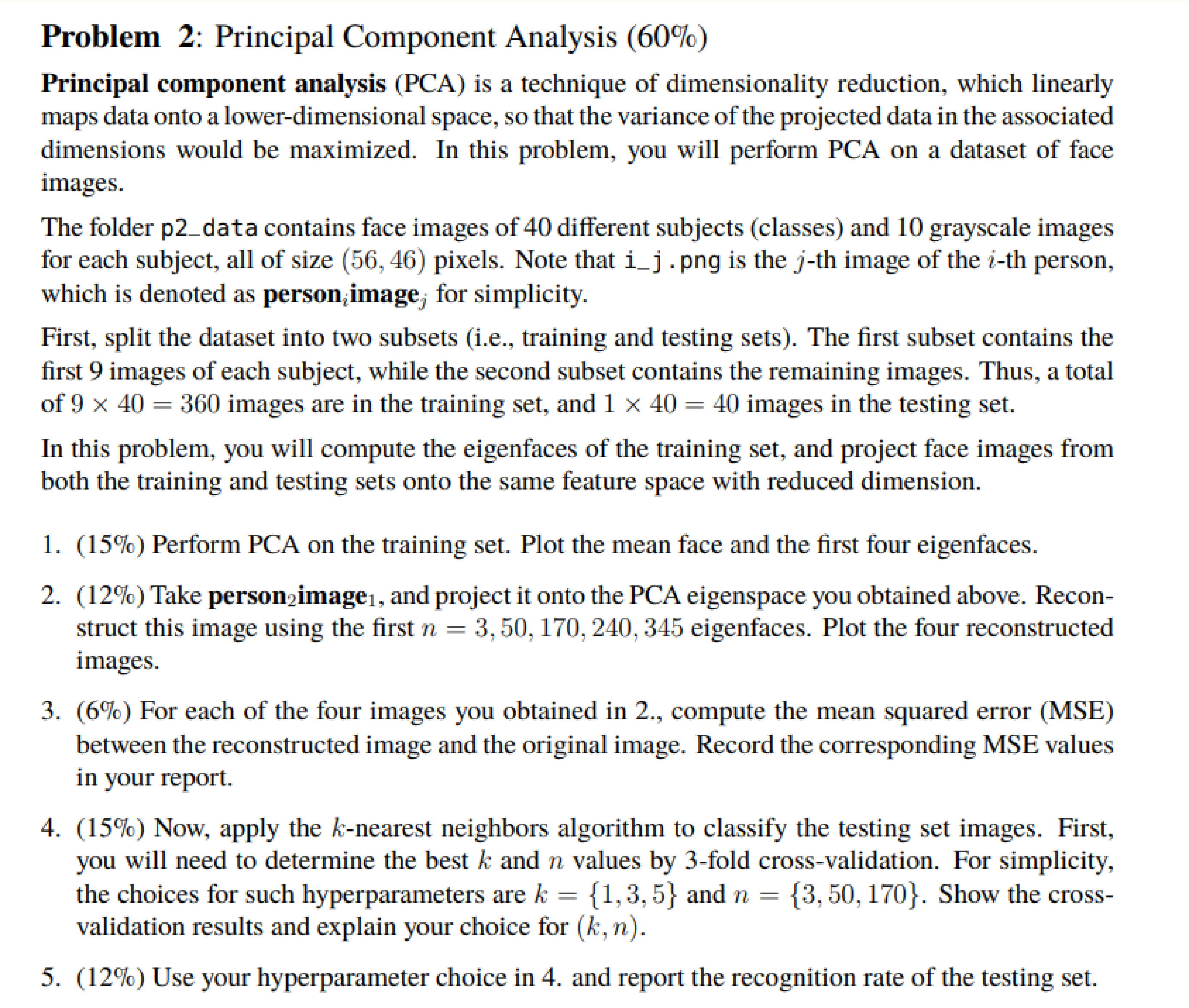


The lab practice referred to NTU Deep Learning for Computer Vision (DLCV) course website.

LAB – FACE RECOGNITION

- Plot the mean face and eigenfaces
- Project face images onto the eigenspace
- Plot the reconstructed image
- Compute mean squared error
- Apply K-nearest neighbors for classification

The lab practice referred to NTU Deep Learning for Computer Vision (DLCV) course website.



Problem 2: Principal Component Analysis (60%)

Principal component analysis (PCA) is a technique of dimensionality reduction, which linearly maps data onto a lower-dimensional space, so that the variance of the projected data in the associated dimensions would be maximized. In this problem, you will perform PCA on a dataset of face images.

The folder `p2_data` contains face images of 40 different subjects (classes) and 10 grayscale images for each subject, all of size $(56, 46)$ pixels. Note that `i_j.png` is the j-th image of the i-th person, which is denoted as **person$_i$image$_j$** for simplicity.

First, split the dataset into two subsets (i.e., training and testing sets). The first subset contains the first 9 images of each subject, while the second subset contains the remaining images. Thus, a total of $9 \times 40 = 360$ images are in the training set, and $1 \times 40 = 40$ images in the testing set.

In this problem, you will compute the eigenfaces of the training set, and project face images from both the training and testing sets onto the same feature space with reduced dimension.

1. (15%) Perform PCA on the training set. Plot the mean face and the first four eigenfaces.
2. (12%) Take **person$_2$image$_1$**, and project it onto the PCA eigenspace you obtained above. Reconstruct this image using the first $n = 3, 50, 170, 240, 345$ eigenfaces. Plot the four reconstructed images.
3. (6%) For each of the four images you obtained in 2., compute the mean squared error (MSE) between the reconstructed image and the original image. Record the corresponding MSE values in your report.
4. (15%) Now, apply the k-nearest neighbors algorithm to classify the testing set images. First, you will need to determine the best k and n values by 3-fold cross-validation. For simplicity, the choices for such hyperparameters are $k = \{1, 3, 5\}$ and $n = \{3, 50, 170\}$. Show the cross-validation results and explain your choice for (k, n).
5. (12%) Use your hyperparameter choice in 4. and report the recognition rate of the testing set.