Face Detection

One of the popular techniques for face recognition is eigenfaces.

And at the heart of eigenfaces is dimensionality reduction technique called principal component analysis (PCA).

PCA

In machine learning, “dimensionality” simply refers to the number of features.

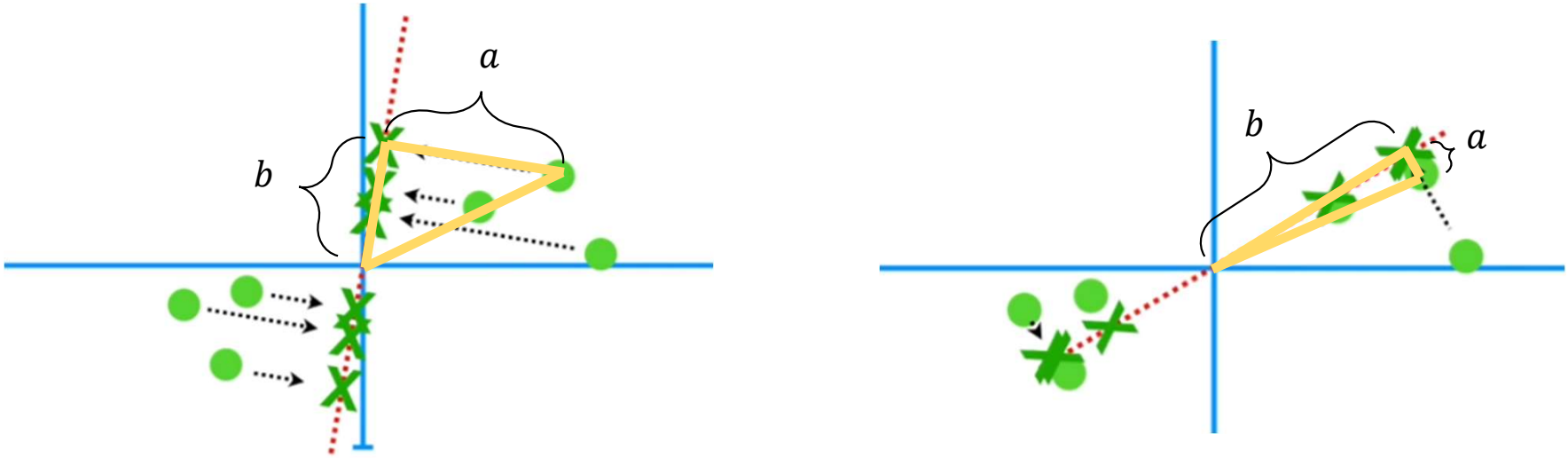
While the performance of any machine learning model increases if we add additional features, at some point a further insertion leads to performance degradation.

Dimensionality reduction is a set of techniques that studies how to shrivel the size of data while preserving the most important information.

PCA is a projection method where data with m-features is projected into a subspace with fewer columns, while retaining the essence of the original data.

PCA

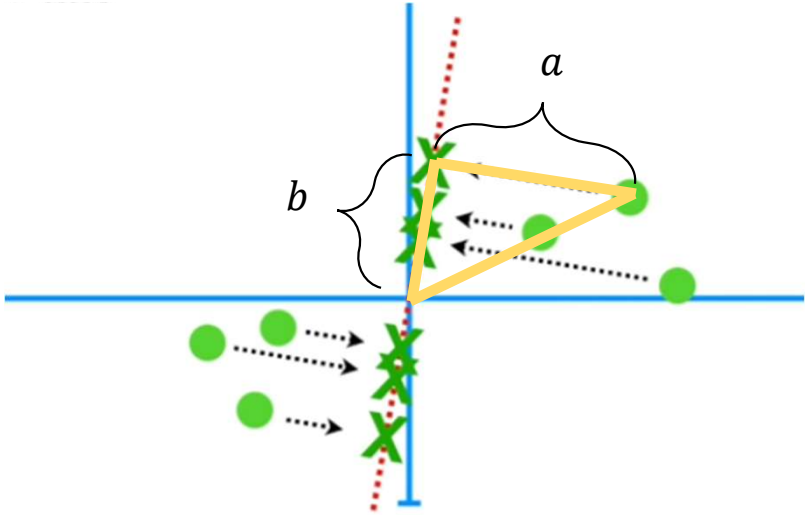
To quantify how good this projecting line fits the data, PCA projects the data onto it.
And try to find the distances from the data to the line that minimizes those distances(a).

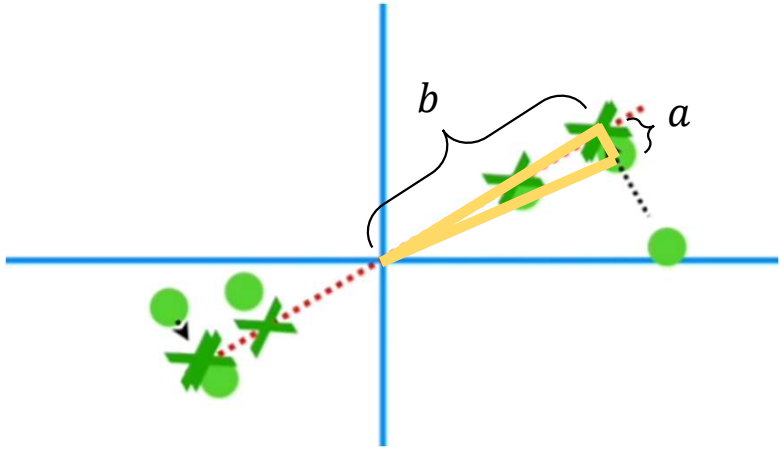


PCA

Using the Pythagorean theorem, it can be changed to maximizing the sum of the squared distances(b) from the projected points to the origin.

It also means we are maximizing the variance (distribution) along the projection axis.





PCA

If $\boldsymbol{X}$ is the original data(standardized, average of each feature is 0), and the sample data projected on vector $\vec{e}$ is

$$X\vec{e} \in \mathbb{R}^{n\times 1}$$

Finding vector that maximizes sample variance of projected data:

$$Var(X\vec{e}) = \frac{1}{n}\sum_{i=1}^{n}(X\vec{e} - E(X\vec{e}))^2 = \frac{1}{n}\sum_{i=1}^{n}(X\vec{e})^2$$

We need to constrain the maximization. The constraint is that $|\vec{e}|^2 = 1$

PCA

Thus, desired vector $\vec{e}$ is an eigenvector of the covariance matrix Σ, and the maximizing vector will be the one associated with the largest eigenvalue λ

$$L = \vec{e}^T \Sigma \vec{e} - \lambda(|\vec{e}|^2 - 1)$$

$$\frac{\partial L}{\partial \vec{e}} = 2\Sigma\vec{e} - 2\lambda\vec{e} = 0$$

$$\Sigma\vec{e} = \lambda\vec{e}$$

PCA

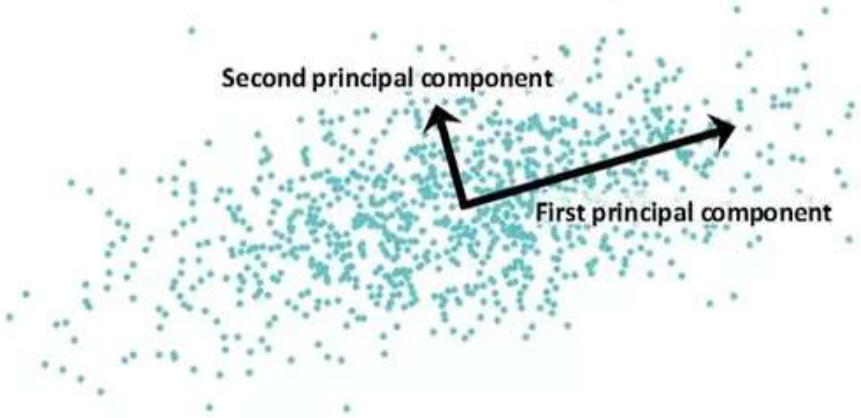
PCA maximizing the variance, minimizing information loss when projecting the data.

$X\vec{e} \in \mathbb{R}^{n\times 1}$ $\Sigma\vec{e} = \lambda\vec{e}$

Σ : The covariance matrix of data $\boldsymbol{X}$

$\vec{e}$: Is an eigenvector of the covariance matrix Σ and the principal components of the data.

λ : Is an eigenvalue of the covariance matrix Σ and it denotes the amount of variability captured along dimension.



PCA

Reorient the data from the original axes to the ones represented by the principal components. $(d \rightarrow p)$

$$y_{n\times p} = X_{n\times d}\mathbf{U}_{d\times p}$$

Therefore any vector in train data can be written as a linear combination of the eigenvectors. $\vec{e}_{\boldsymbol{i}}$

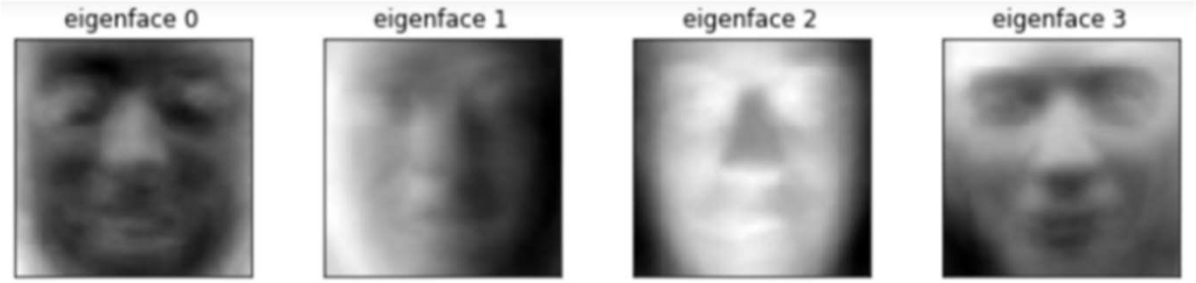
Removing correlated features and greatly reduce the number of dimensions.

Eigenface

1. Transform the images into vectors for PCA.

2. Then, find the mean and subtract it from data.

3. Create covariance matrix for the training set and calculate eigenvalues and eigenvectors.



Resulting eigenvectors are called eigenfaces.

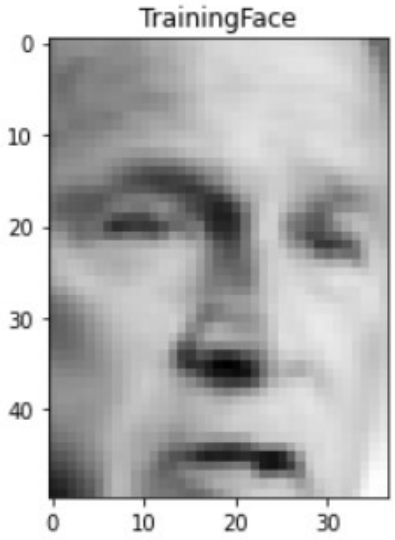
Eigenface

Eigenfaces are blurry depictions of faces that each highlight a certain type of feature.

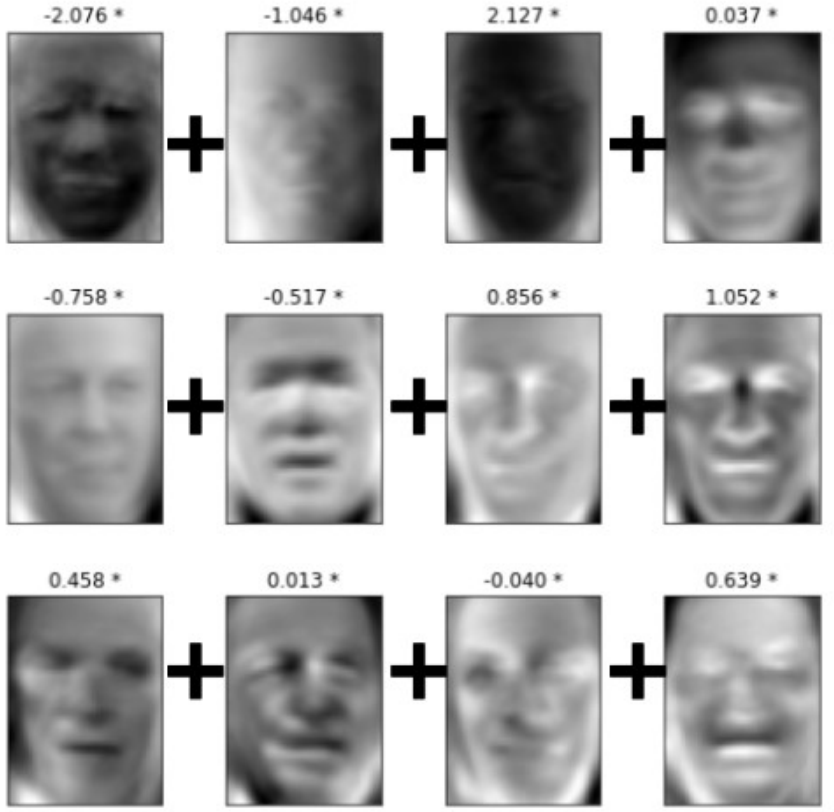
Eigenfaces are eigenvectors extracted from train data sets of images of human faces.

Someone’s face can be reconstructed from a suitable linear combination of eigenfaces.

Eigenface


TrainingFace
0
10
20
30
40
0
10
20
30

=

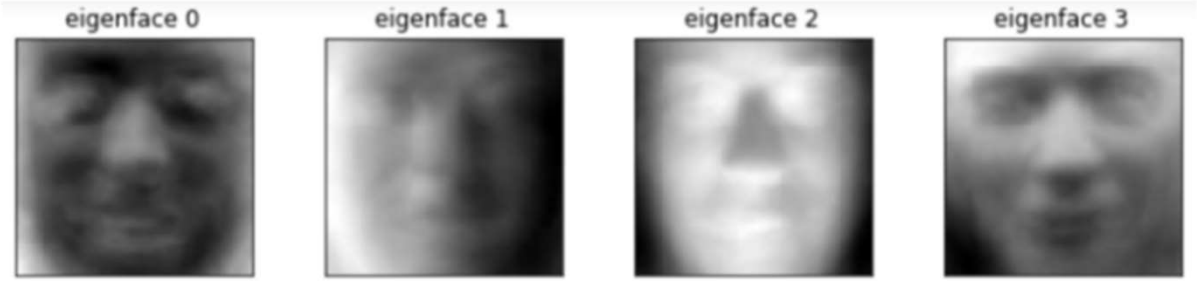

-2.076 *
-1.046 *
2.127 *
0.037 *
-0.758 *
-0.517 *
0.856 *
1.052 *
0.458 *
0.013 *
-0.040 *
0.639 *

Face Recognition

1. Transform the **train** images into centered vectors for PCA.

2. Calculate the eigenvectors and eigenvalues of the covariance matrix.

3. Find the optimal transformation matrix by selecting the principal components.
   (principal components = some eigenvectors = some eigenfaces)



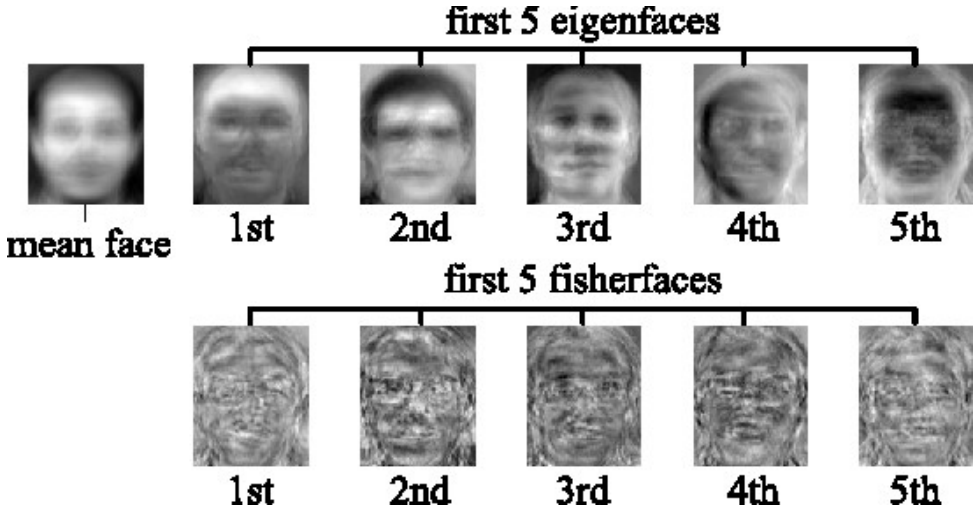
Face Recognition

4. Calculate eigenface of the centered **test** data.
5. Find the nearest neighbor to a set of projected training images.

	George	Jeff	John	Tom
distanc-e	252.996	253.641	47.8731	158.434

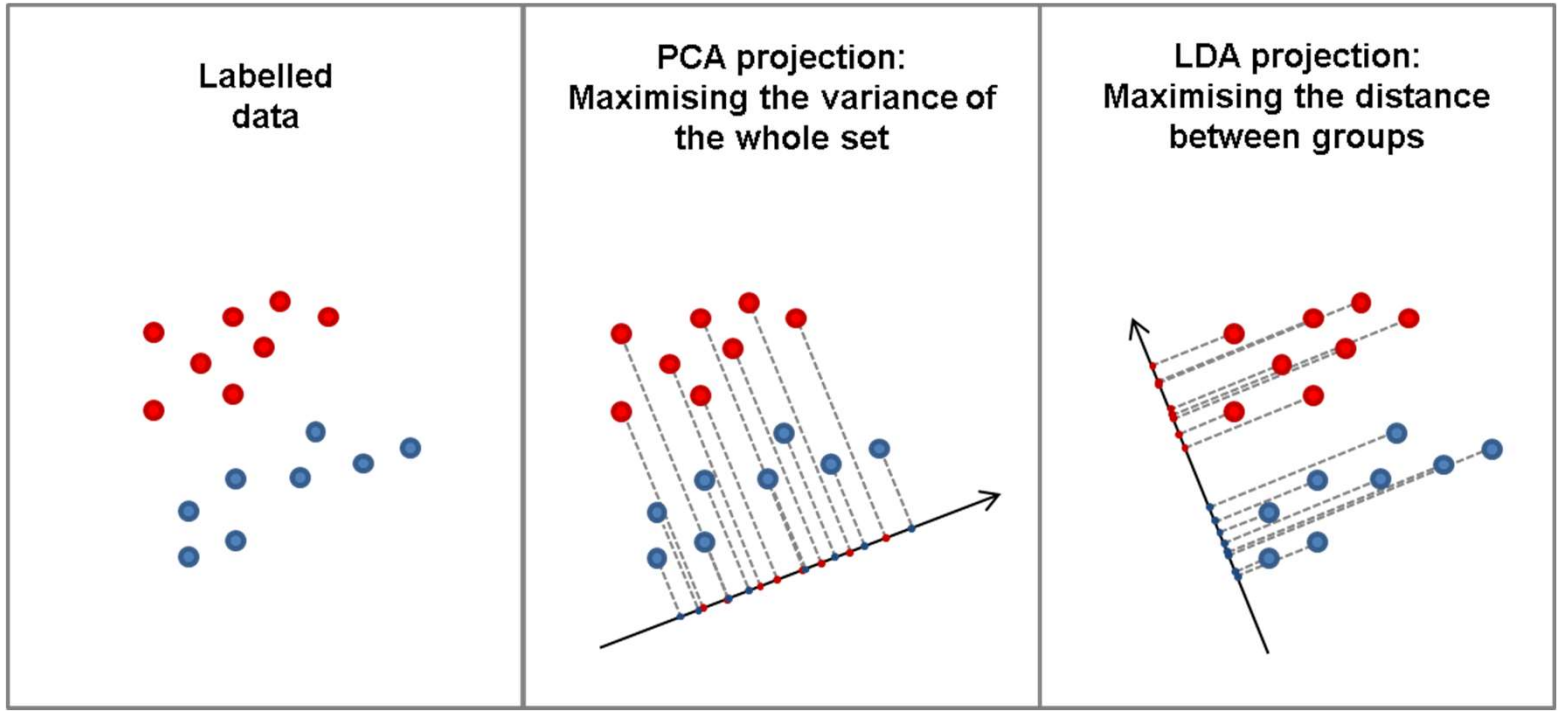
Face Recognition

Eigenface method of face recognition was improved upon using linear discriminant analysis (LDA) to produce Fisherfaces.



LDA

LDA seeks to reduce dimensionality while preserving as much of the class discriminatory information as possible



LDA

Define the scatter, an equivalent of the variance, as;

- sum of square differences between the projected samples and their mean

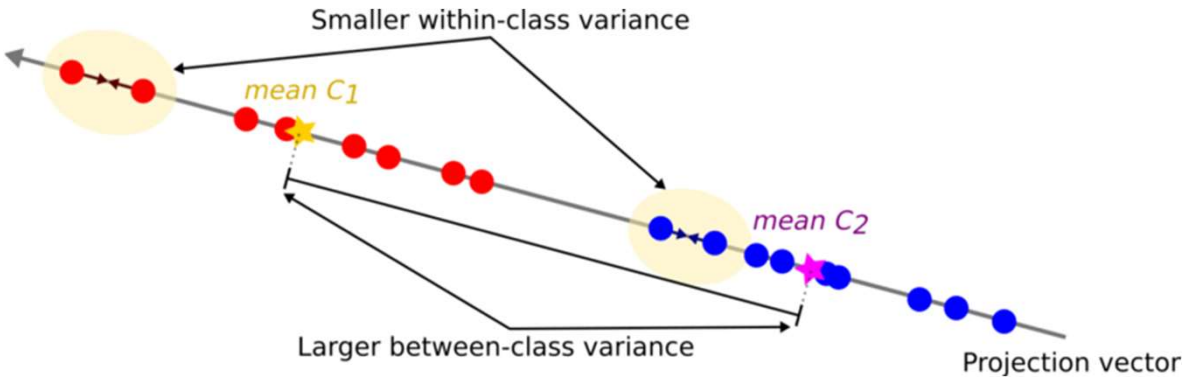
Multi-class LDA is based on the analysis of two scatter matrices:

- within-class scatter matrix

$$\mathbf{S}_w = \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu}_{y_i})(\mathbf{x}_i - \boldsymbol{\mu}_{y_i})^T$$

- between-class scatter matrix.

$$\mathbf{S}_b = \sum_{k=1}^{m} n_k (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^T$$



LDA

We can finally express the Fisher criterion in terms of S_W and S_B as

$$J(w) = \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} = \frac{w^T S_B w}{w^T S_W w}$$

Solving the generalized eigenvalue problem

$$S_B{}^{-1} S_W w = \lambda w$$

Using the same notation as PCA, the solution will be the eigenvector(s) of $S_B{}^{-1} S_W$

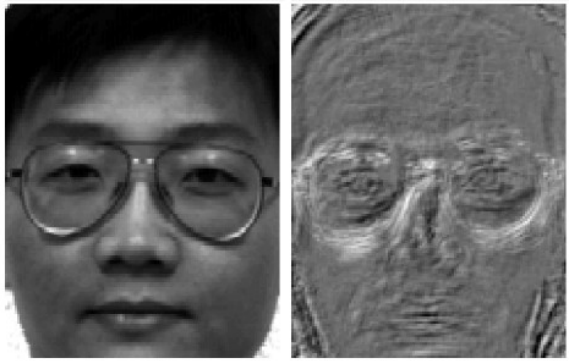
Resulting eigenvectors are called fisherfaces.

Fisherface

Eigenfaces relies on PCA, Fisherfaces relies on LDA for dimensionality reduction.

Fisherfaces algorithm extracts principle components that separates one individual from another.

The Fisherface is especially useful when facial images have large variations in illumination and facial expression.



Fisherface

Fischerfaces yields much better face recognition performance than eigenfaces.

However, due to the need of better classification, the dimension of projection in face space is not as compact as Eigenface.

So it loses the ability to reconstruct faces because the Eigenspace is lost.