《模式识别》

第四章 归一化、判别分析、人脸识别

马锦华

https://cse.sysu.edu.cn/teacher/MaJinhua

SUN YAT-SEN University

中山大學 计算机学院
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING, SYSU

课程目录（暂定）

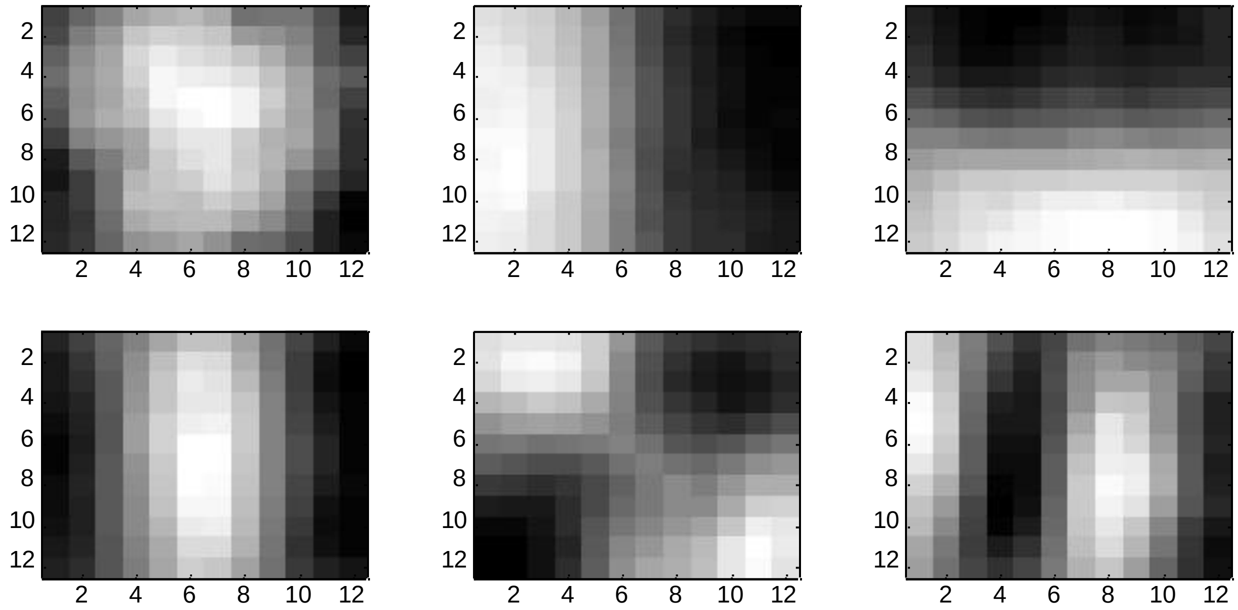
第一章	课程简介与预备知识	6学时
第二章	特征提取与表示	6学时
第三章	主成分分析	3学时
第四章	归一化、判别分析、人脸识别	3学时
第五章	EM算法与聚类	3学时
第六章	贝叶斯决策理论	3学时
第七章	线性分类器与感知机	3学时
第八章	支持向量机	3学时
第九章	神经网络、正则项和优化方法	3学时
第十章	卷积神经网络及经典框架	3学时
第十一章	循环神经网络	3学时
第十二章	Transformer	3学时
第十三章	自监督与半监督学习	3学时
第十四章	开放世界模式识别	6学时

Let’s recap

- **PCA**

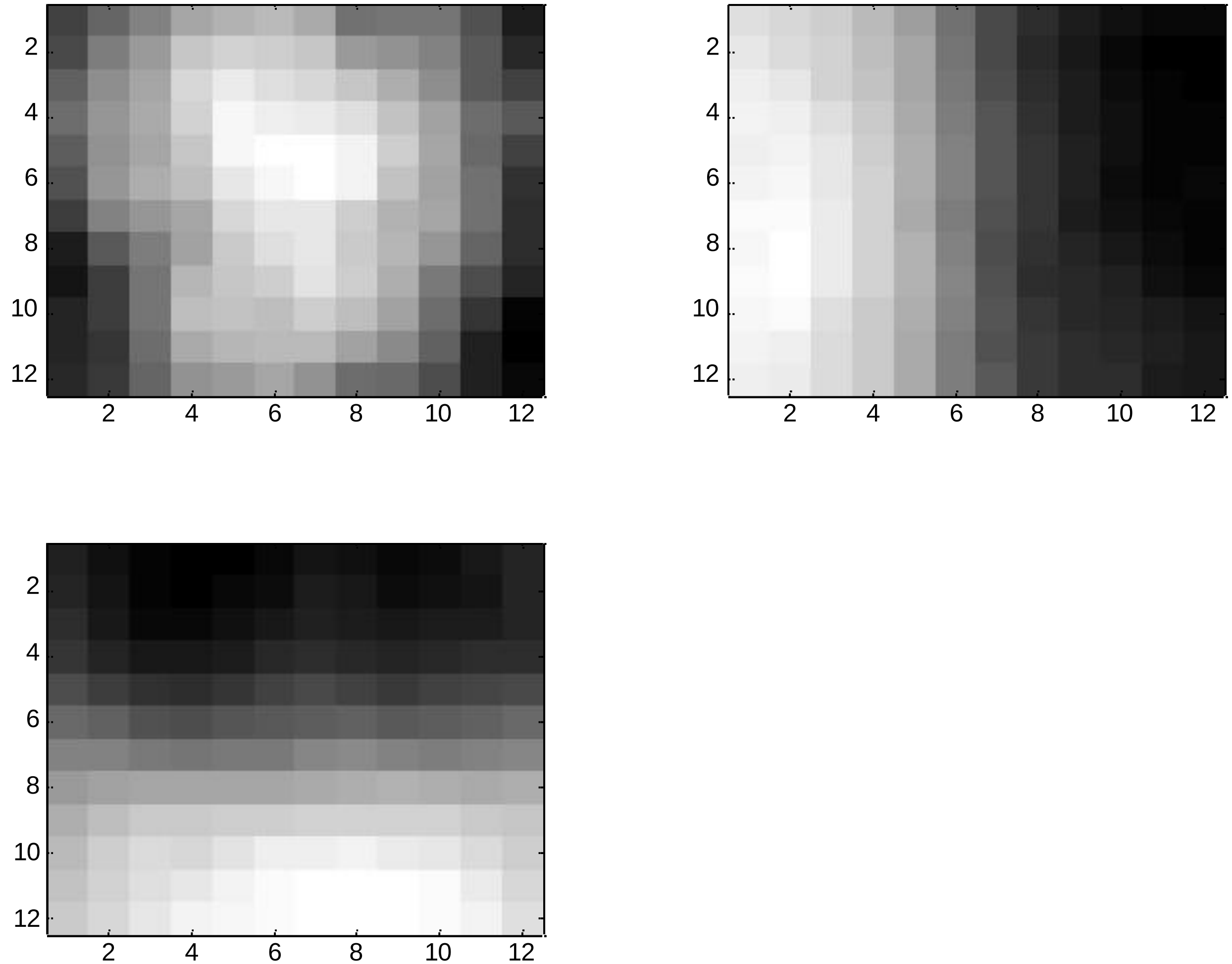
PCA compression: 144D -> 6D

6 most important eigenvectors



PCA compression: 144D) 3D

3 most important eigenvectors



What we will learn today

- **Feature normalization**
- Introduction to face recognition
- A simple recognition pipeline with kNN
- The Eigenfaces Algorithm
- Linear Discriminant Analysis (LDA)

Turk and Pentland, Eigenfaces for Recognition, *Journal of Cognitive Neuroscience* **3** (1): 71–86, 1991.

P. Belhumeur, J. Hespanha, and D. Kriegman. "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". *IEEE Transactions on pattern analysis and machine intelligence* **19** (7): 711. 1997.

每维度规范化

- per-dimension normalization
  - 虚拟的例子（判别性别）
    - 假设用两个特征：身高和体重
    - 如果1. 身高单位毫米，体重单位吨，那么？
    - 如果2. 身高单位公里，体重单位克，那么？
  - 很多时候，不同的维度需要统一到同样的取值范围！
- 训练集：$\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$, $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \dots, x_{id})$
  - 对每一维j，其数据为$x_{1j}, x_{2j}, \dots, x_{nj}$
  - 取其最小值 $x_{min,j}$和最大值$x_{max,j}$
  - 对这一维的任何数据 $x_{ij} \leftarrow \frac{x_{ij}-x_{min,j}}{x_{max,j}-x_{min,j}}$

稀疏数据

- 新数据的范围是？各维度统一了吗？
  - [0 1] （训练集中的情况）
  - 若某一维$x_{max,j} = x_{min,j}$？
  - 也可以统一到[-1 1]

$$x_{ij} \leftarrow 2 \times \left(\frac{x_{ij} - x_{min,j}}{x_{max,j} - x_{min,j}} - 0.5 \right)$$

- 稀疏数据sparse data：数据中很多维度值为0
  - 如果所有数据≥ 0，在两种归一化中，原来是0的会变成什么？

ℓ_2或ℓ_1归一化

- 若各维度取值范围的不同是有意义的，但是不同数据点之间的大小（如向量长度norm）应保持一致
  - 对每个数据$\boldsymbol{x}_i = (x_{i1}, x_{i2}, \dots, x_{id})$

$$x_{ij} \leftarrow \frac{x_{ij}}{\|\boldsymbol{x}_i\|_{\ell_2}} \qquad \|\boldsymbol{x}_i\|_{\ell_2} = \sqrt{\boldsymbol{x}_i^T \boldsymbol{x}_i}$$

- ℓ_1归一化
  - 适用于非负的特征，即$x_{ij} \geq 0$总成立
  - 若数据$\boldsymbol{x}_i$是直方图(histogram)时，经常是最佳的

$$x_{ij} \leftarrow \frac{x_{ij}}{\|\boldsymbol{x}_i\|_{\ell_1}} \qquad \|\boldsymbol{x}_i\|_{\ell_1} = \sum_{j=1}^{d} |x_{ij}|$$

zero-mean, unit variance

- 有时候有理由相信每一个维度是服从高斯分布的
  - 希望每一个维度归一化到$N(0,1)$
- 对每一维j，其数据为$x_{1j}, x_{2j}, \ldots, x_{nj}$
  - 计算其均值$\hat{\mu}_j$和方差$\hat{\sigma}_j^2$
  - 对每一个特征值

$$x_{ij} \leftarrow \frac{x_{ij} - \hat{\mu}_j}{\hat{\sigma}_j}$$

规范化测试数据

- 怎样归一化测试数据？
  - 从测试集寻找最大值、最小值、均值？
- 除了在测试的时候，永远不要使用测试数据！
  - 测试集和训练集应该使用相同的归一化方法
    - 还记得吗？训练和测试集应该从相同的$p(\boldsymbol{x})$取样
    - 同样的归一化会保持这个限定！
  - 这个原则同样适用于交叉验证！
- 那么，怎样做？
  - 保存从训练集上取得的归一化参数(parameter)
  - 使用同样的公式和保存的参数来归一化测试集

小结

- 归一化的方法应该是根据数据的特点来选择的
  - 在做任何机器学习之前，先搞清你的数据的特点
    - 稀疏？
    - 每一维有没有含义？
    - 每一维里面值的分布情况？Gauss？
    - 看你的数据！Do visualization!
- 归一化可能对准确度有极大的影响！
  - 在有些例子里，正确的归一化能大幅度提高 accuracy
- 不同的归一化方法可以混合使用

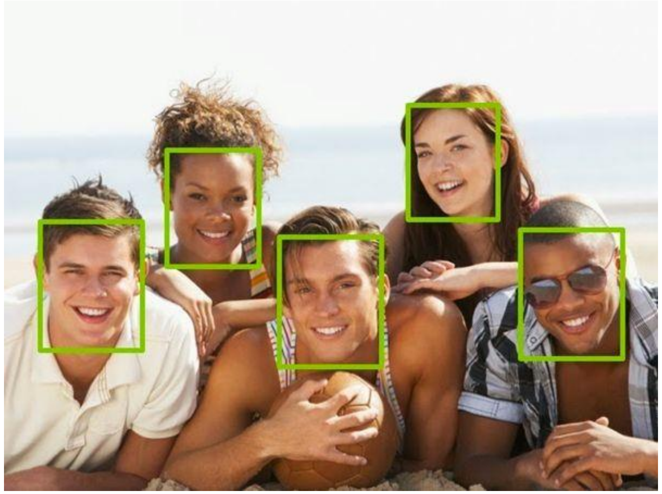
What we will learn today

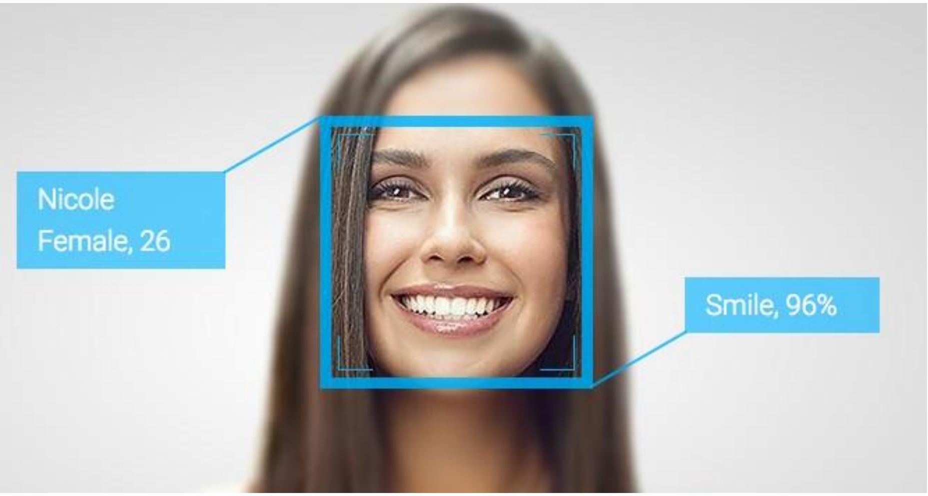
- Feature normalization
- **Introduction to face recognition**
- A simple recognition pipeline with kNN
- The Eigenfaces Algorithm
- Linear Discriminant Analysis (LDA)

Turk and Pentland, Eigenfaces for Recognition, *Journal of Cognitive Neuroscience* **3** (1): 71–86, 1991.

P. Belhumeur, J. Hespanha, and D. Kriegman. "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". *IEEE Transactions on pattern analysis and machine intelligence* **19** (7): 711. 1997.

Detection versus Recognition



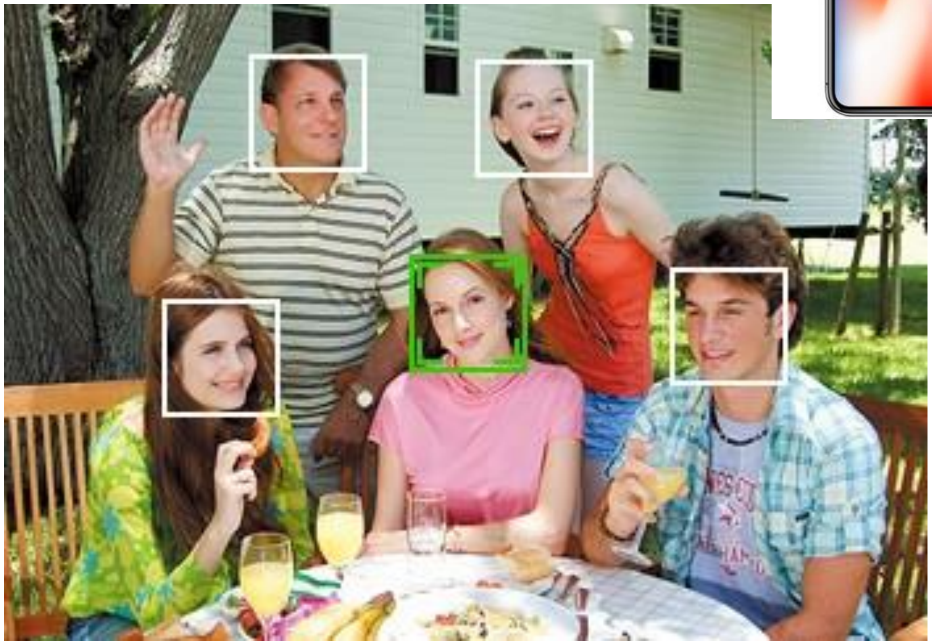


Detection finds the faces in images

Recognition recognizes WHO the person is

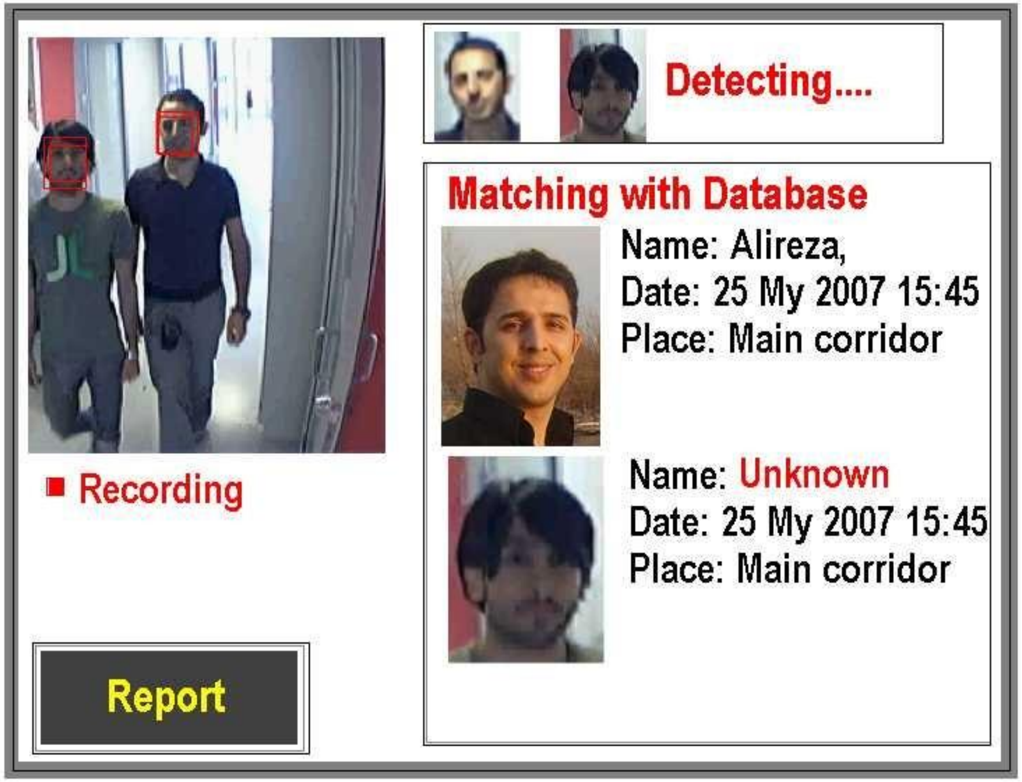
Face Recognition

- Digital photography



Face Recognition

- Digital photography
- Surveillance

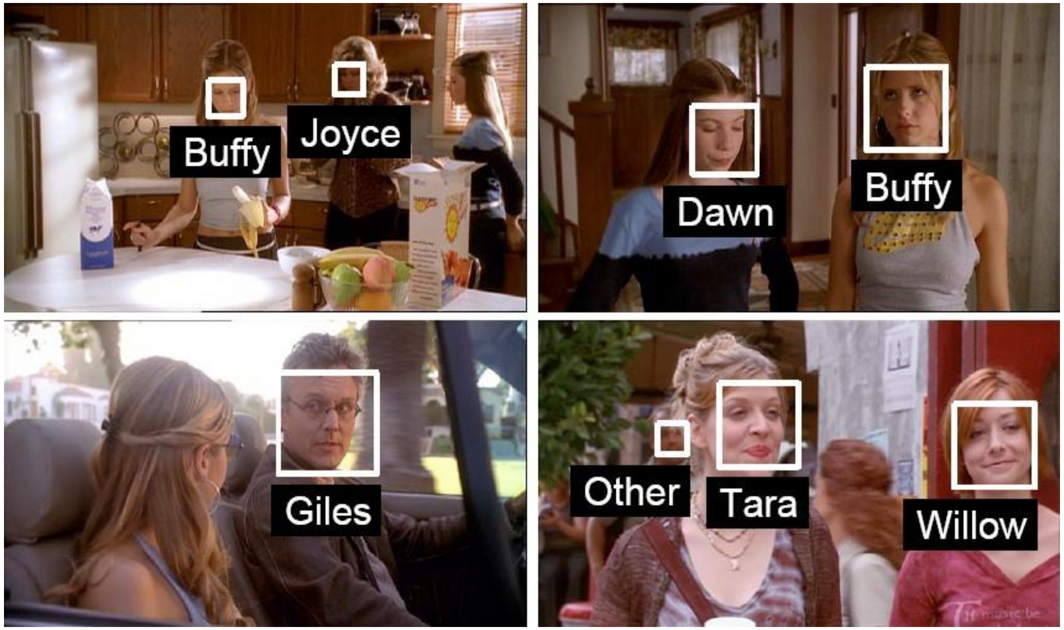


Face Recognition

- Digital photography
- Surveillance
- Album organization

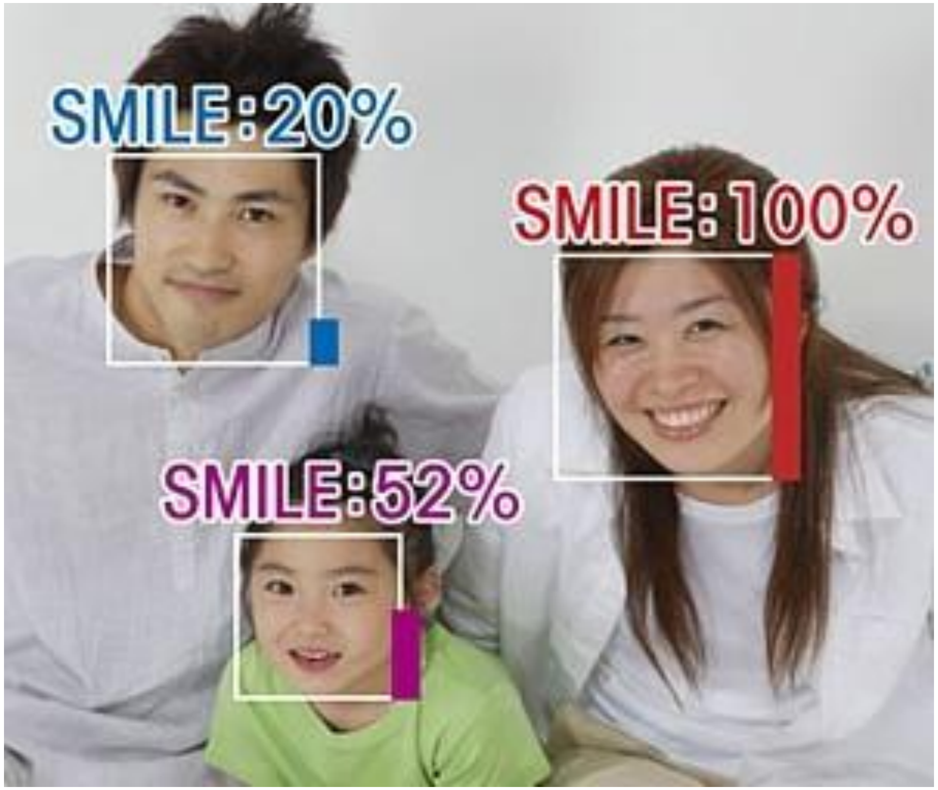
Face Recognition

- Digital photography
- Surveillance
- Album organization
- Person tracking/id.



Face Recognition

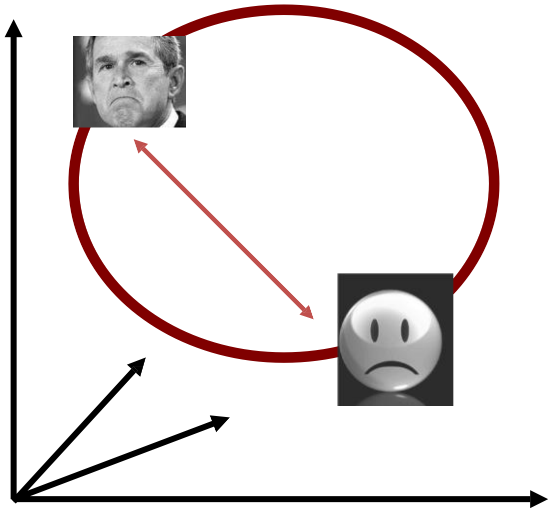
- Digital photography
- Surveillance
- Album organization
- Person tracking/id.
- Emotions and expressions



Face Recognition

- Digital photography
- Surveillance
- Album organization
- Person tracking/id.
- Emotions and expressions
- Security/warfare
- Tele-conferencing
- Etc.

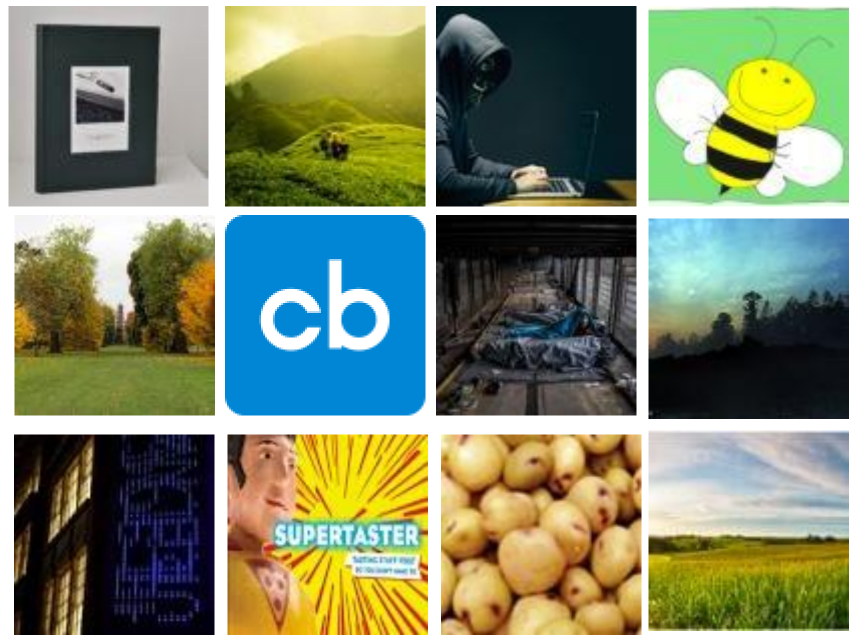
The Space of Faces



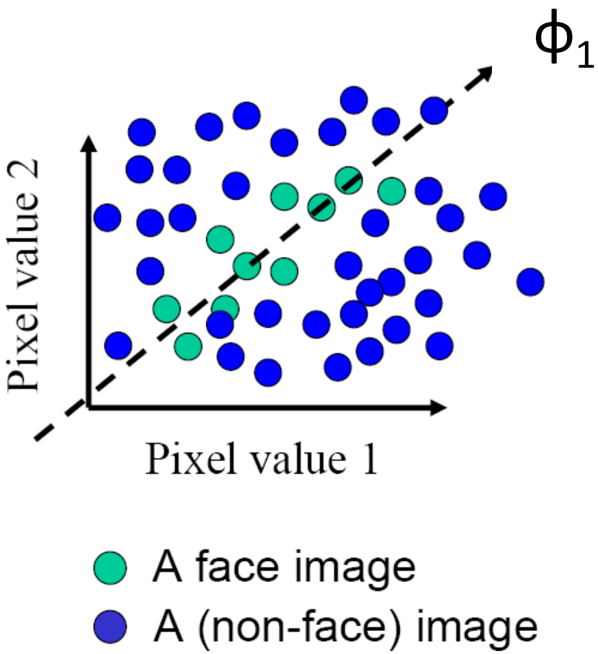
- An image is a point in a high dimensional space
  - If represented in grayscale intensity, an N x M image is a point in R^{NM}
  - E.g. 100x100 image = 10,000 dim

Slide credit: Chuck Dyer, Steve Seitz, Nishino

100x100 images can contain many things other than faces!



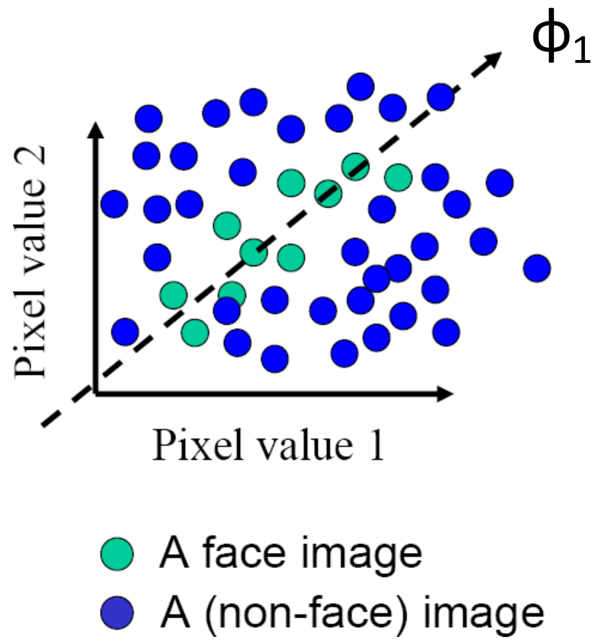
The Space of Faces

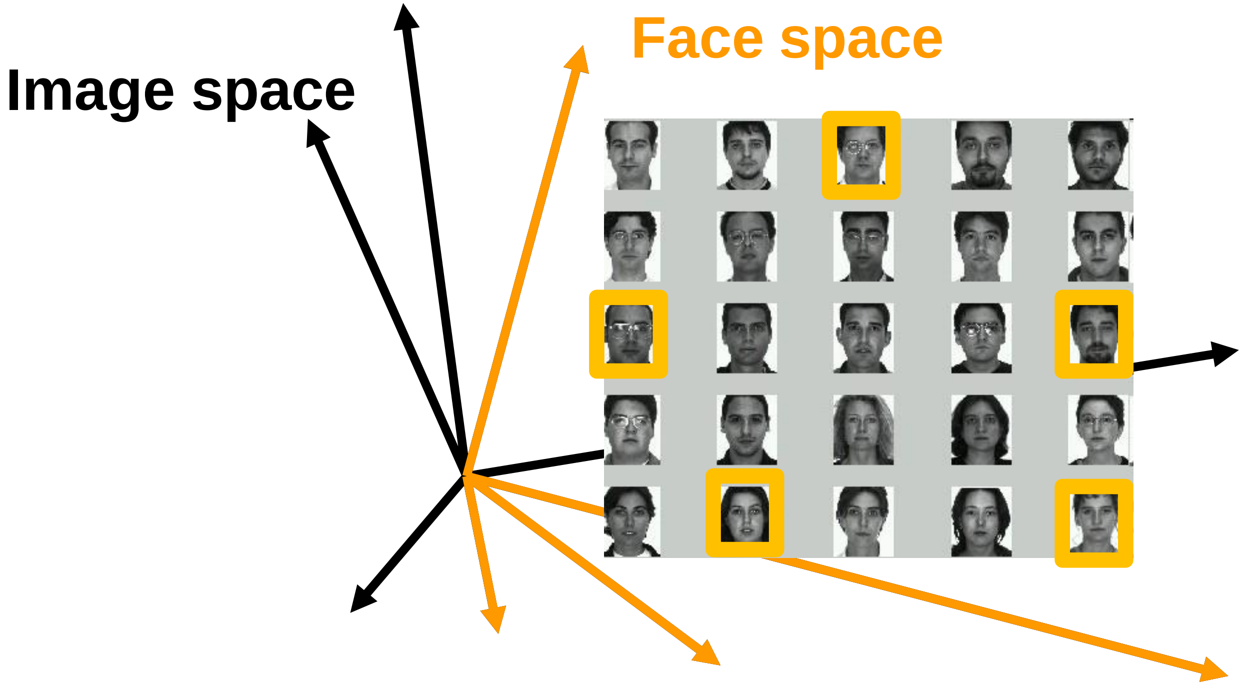


- An image is a point in a high dimensional space
  - If represented in grayscale intensity, an N x M image is a point in R^{NM}
  - E.g. 100x100 image = 10,000 dim
- However, relatively few high dimensional vectors correspond to valid face images
- We want to effectively model the subspace of face images

Slide credit: Chuck Dyer, Steve Seitz, Nishino

Where have we seen something like this before?





•Compute n-dim subspace such that the projection of the data points onto the subspace has the largest variance among all n-dim subspaces.

• Maximize the scatter of the training images in face space

Key Idea

- So, compress them to a low-dimensional subspace that captures key appearance characteristics of the visual DOFs.

- USE PCA for estimating the sub-space (dimensionality reduction)

- Compare two faces by projecting the images into the subspace and measuring the EUCLIDEAN distance between them.

What we will learn today

- Feature normalization
- Introduction to face recognition
- **A simple recognition pipeline with kNN**
- The Eigenfaces Algorithm
- Linear Discriminant Analysis (LDA)

Turk and Pentland, Eigenfaces for Recognition, *Journal of Cognitive Neuroscience* **3** (1): 71–86, 1991.

P. Belhumeur, J. Hespanha, and D. Kriegman. "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". *IEEE Transactions on pattern analysis and machine intelligence* **19** (7): 711. 1997.

Object recognition:

a classification framework

- Apply a prediction function to a feature representation of the image to get the desired output:

f() = “apple”

f() = “tomato”

f() = “cow”

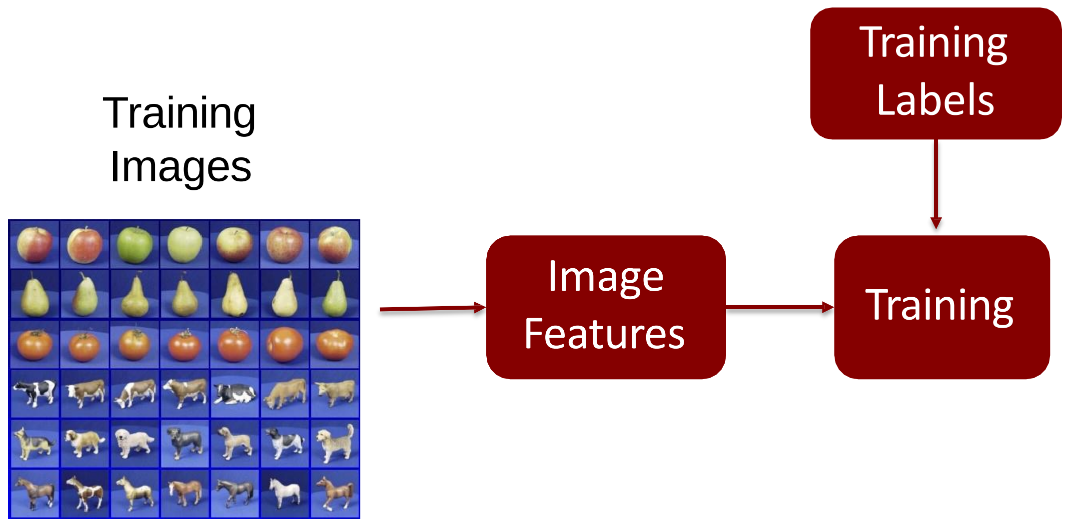
Dataset: ETH-80, by B. Leibe   Slide credit: L. Lazebnik

A simple pipeline - Training

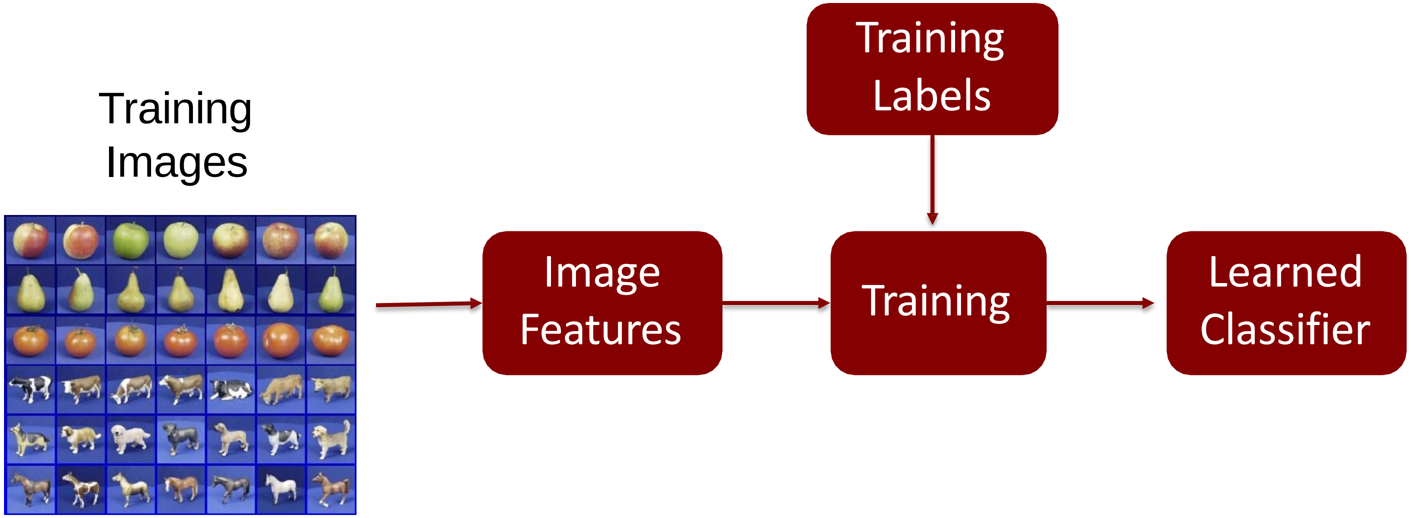
Training
Images

Image
Features

A simple pipeline - Training



A simple pipeline - Training



A simple pipeline - Testing

Training Images

Image Features

Training Labels

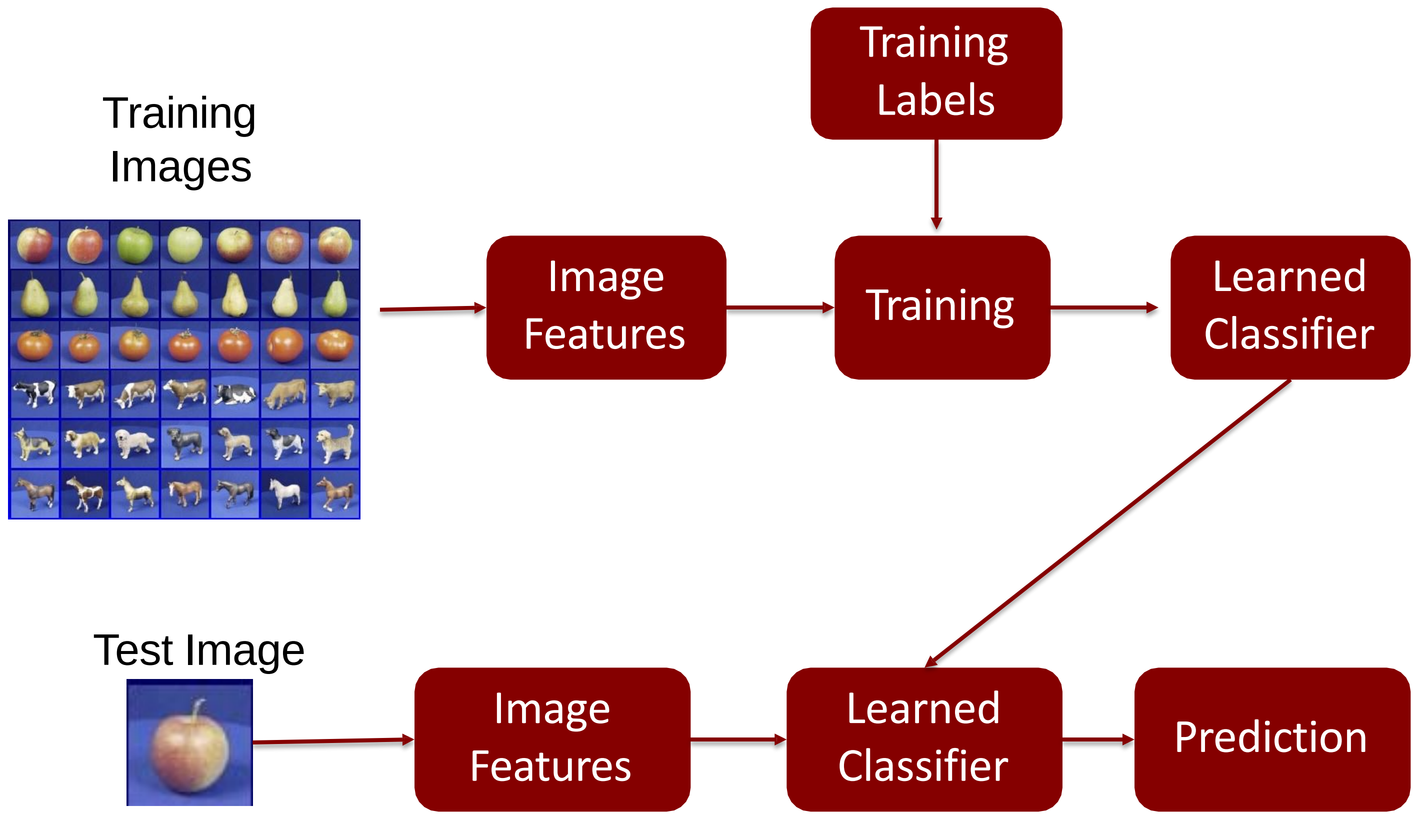
Training

Learned Classifier

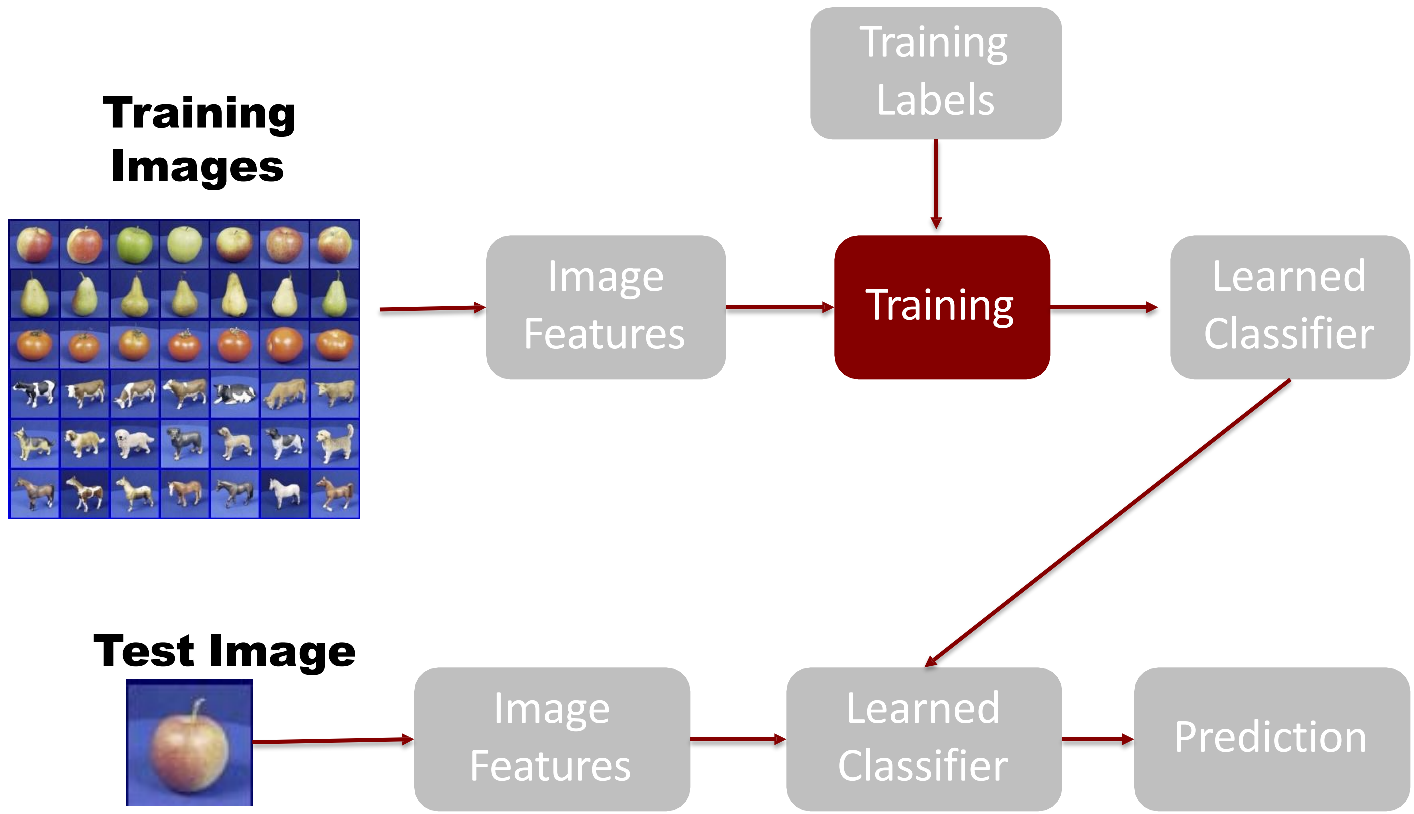
Test Image

Image Features

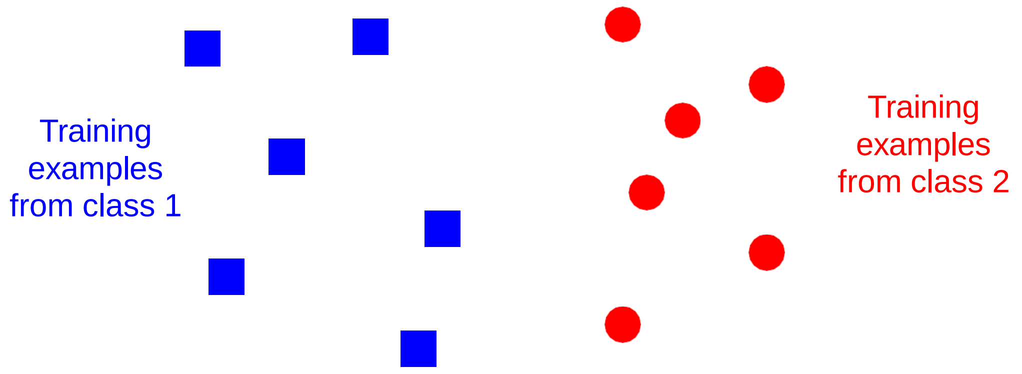
A simple pipeline - Testing



A simple pipeline - Training

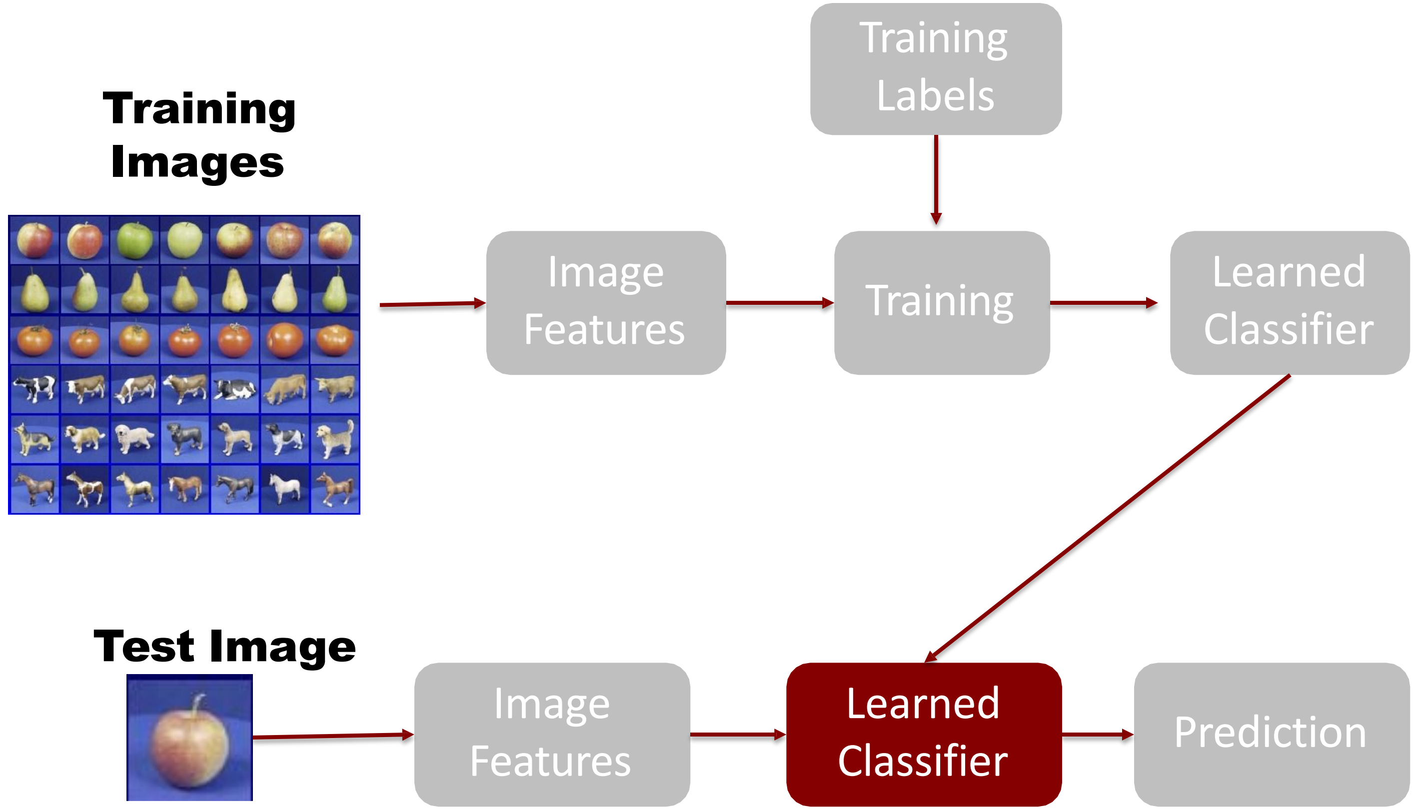


Classifiers: Nearest neighbor

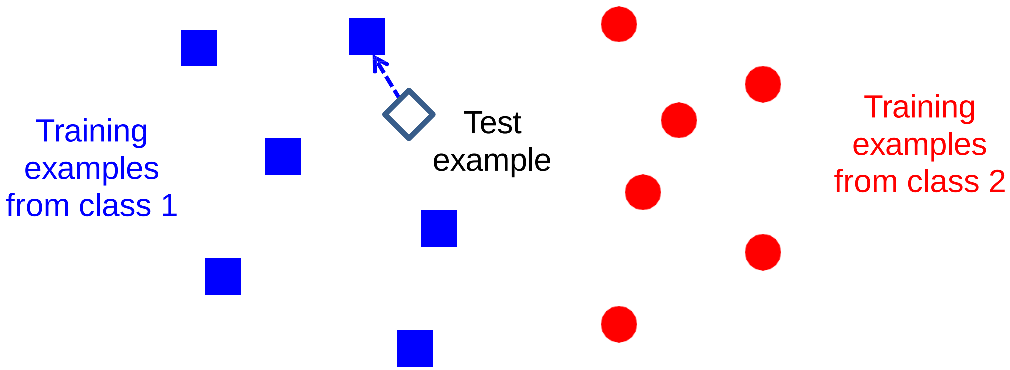


Slide credit: L. Lazebnik

A simple pipeline - Testing



Classifiers: Nearest neighbor



Slide credit: L. Lazebnik

What we will learn today

- Feature normalization
- Introduction to face recognition
- A simple recognition pipeline with kNN
- **The Eigenfaces Algorithm**
- Linear Discriminant Analysis (LDA)

Turk and Pentland, Eigenfaces for Recognition, *Journal of Cognitive Neuroscience* **3** (1): 71–86, 1991.

P. Belhumeur, J. Hespanha, and D. Kriegman. "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". *IEEE Transactions on pattern analysis and machine intelligence* **19** (7): 711. 1997.

Eigenfaces: key idea

- Assume that most face images lie on a low-dimensional subspace determined by the first k (k<<d) directions of maximum variance
- Use PCA to determine the vectors or “eigenfaces” that span that subspace
- Represent all face images in the dataset as linear combinations of eigenfaces

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Training images: $\mathbf{x}_1,\ldots,\mathbf{x}_N$

Eigenface algorithm

- Training

1. Align training images $x_1, x_2, \ldots, x_N$

Note that each image is formulated into a long vector!

2. Compute average face

$$\mu = \frac{1}{N}\sum x_i$$

3. Compute the difference image (the <u>centered data matrix</u>)

$$X_c = \begin{bmatrix} | & & | \\ x_1 & \ldots & x_n \\ | & & | \end{bmatrix} - \begin{bmatrix} | & & | \\ \mu & \ldots & \mu \\ | & & | \end{bmatrix}$$

$$= X - \mu 1^T = X - \frac{1}{n} X 1 1^T = X\left(I - \frac{1}{n} 1 1^T\right)$$

Eigenface algorithm

4. Compute the covariance matrix

$$\boxed{\Sigma = \frac{1}{n}\begin{bmatrix} | & & | \\ x_1^c & \dots & x_n^c \\ | & & | \end{bmatrix}\begin{bmatrix} - & x_1^c & - \\ & \vdots & \\ - & x_n^c & - \end{bmatrix} = \frac{1}{n} X_c X_c^T}$$

5. Compute the eigenvectors of the covariance matrix Σ
6. Compute each training image x_i 's projections as

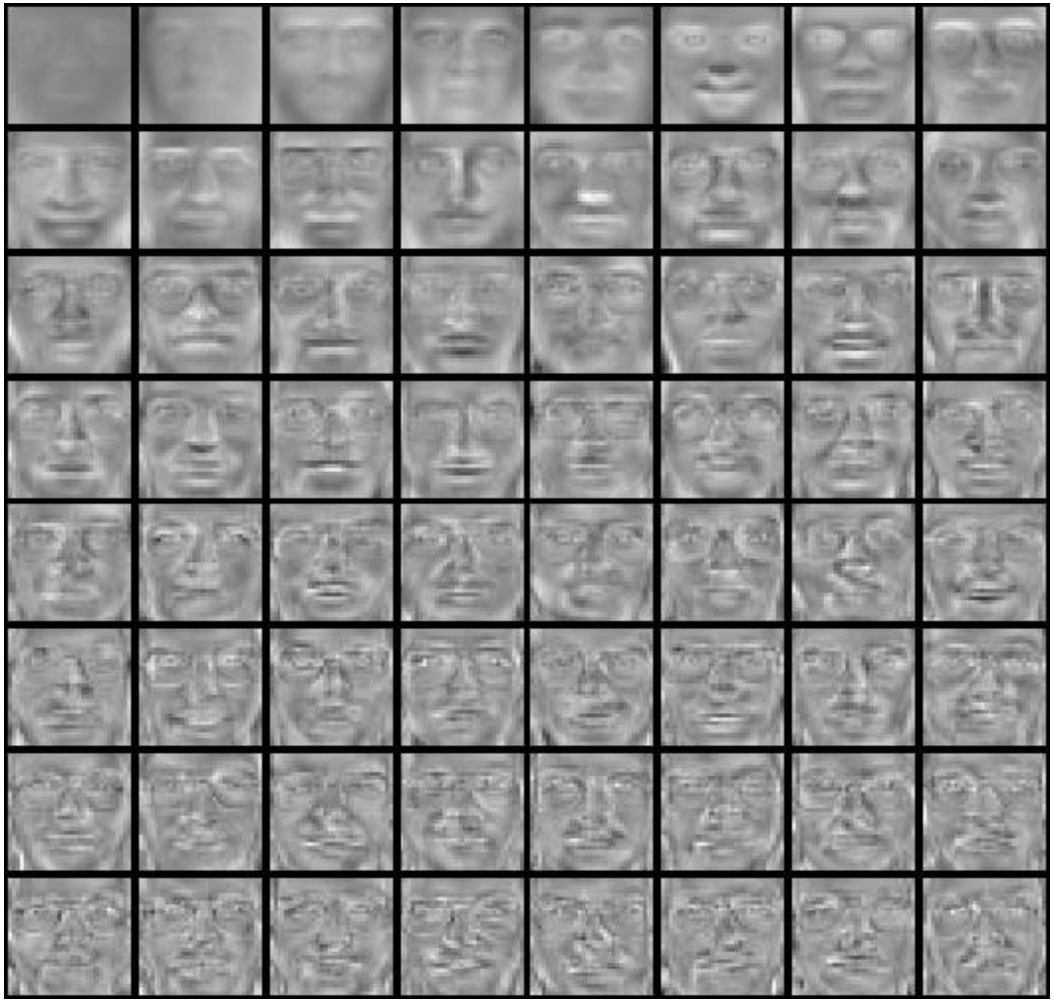
$$x_i \rightarrow \left(x_i^c \cdot \varphi_1, x_i^c \cdot \varphi_2, \ldots, x_i^c \cdot \varphi_K\right) \equiv \left(a_1, a_2, \ldots, a_K\right)$$

7. Visualize the estimated training face x_i

$$x_i \approx \mu + a_1\varphi_1 + a_2\varphi_2 + \ldots + a_K\varphi_K$$

Top eigenvectors: $\Phi_1, \ldots, \Phi_k$

Mean: μ



Eigenface algorithm

x_i

$a_1\varphi_1$ $a_2\varphi_2$ … $a_K\varphi_K$

Reconstructed training face

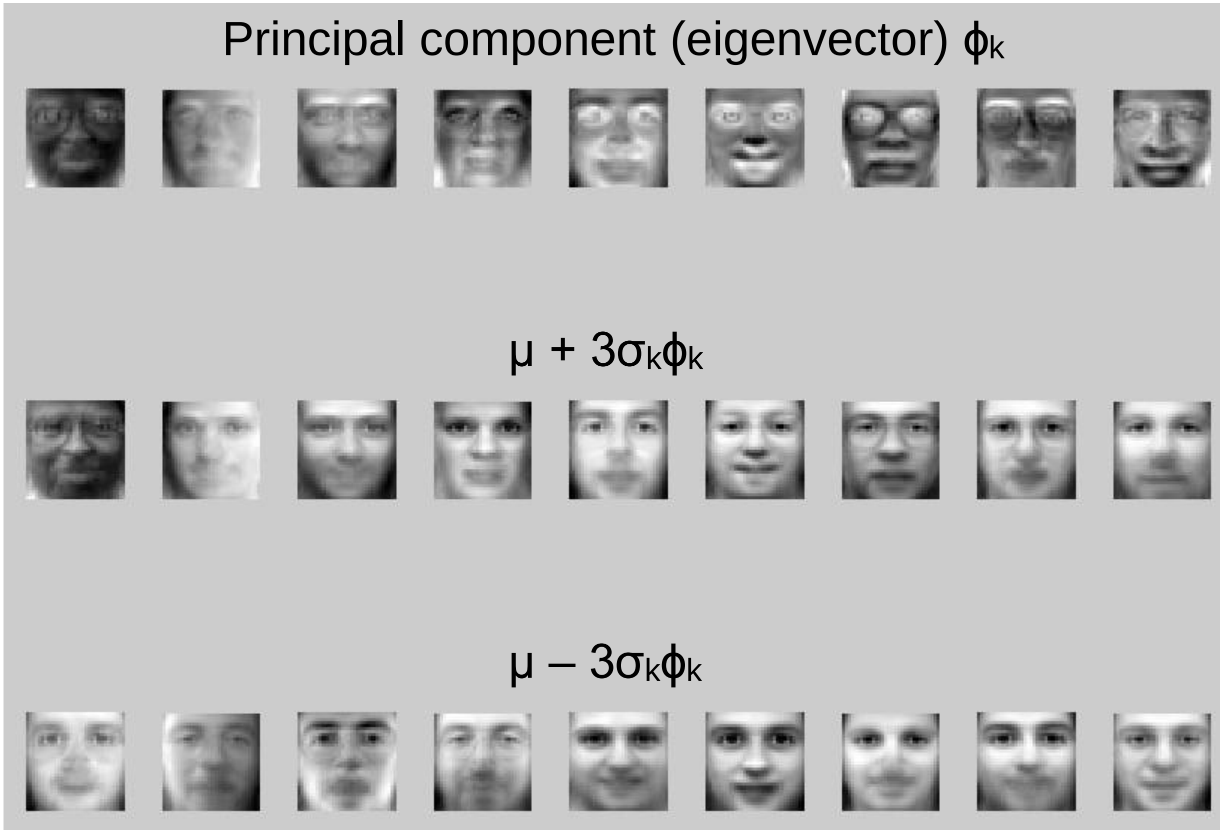
6. Compute each training image x_i 's projections as

$$x_i \rightarrow \left(x_i^c \cdot \varphi_1,\ x_i^c \cdot \varphi_2, \ldots,\ x_i^c \cdot \varphi_K\right) \equiv \left(a_1, a_2, \ldots, a_K\right)$$

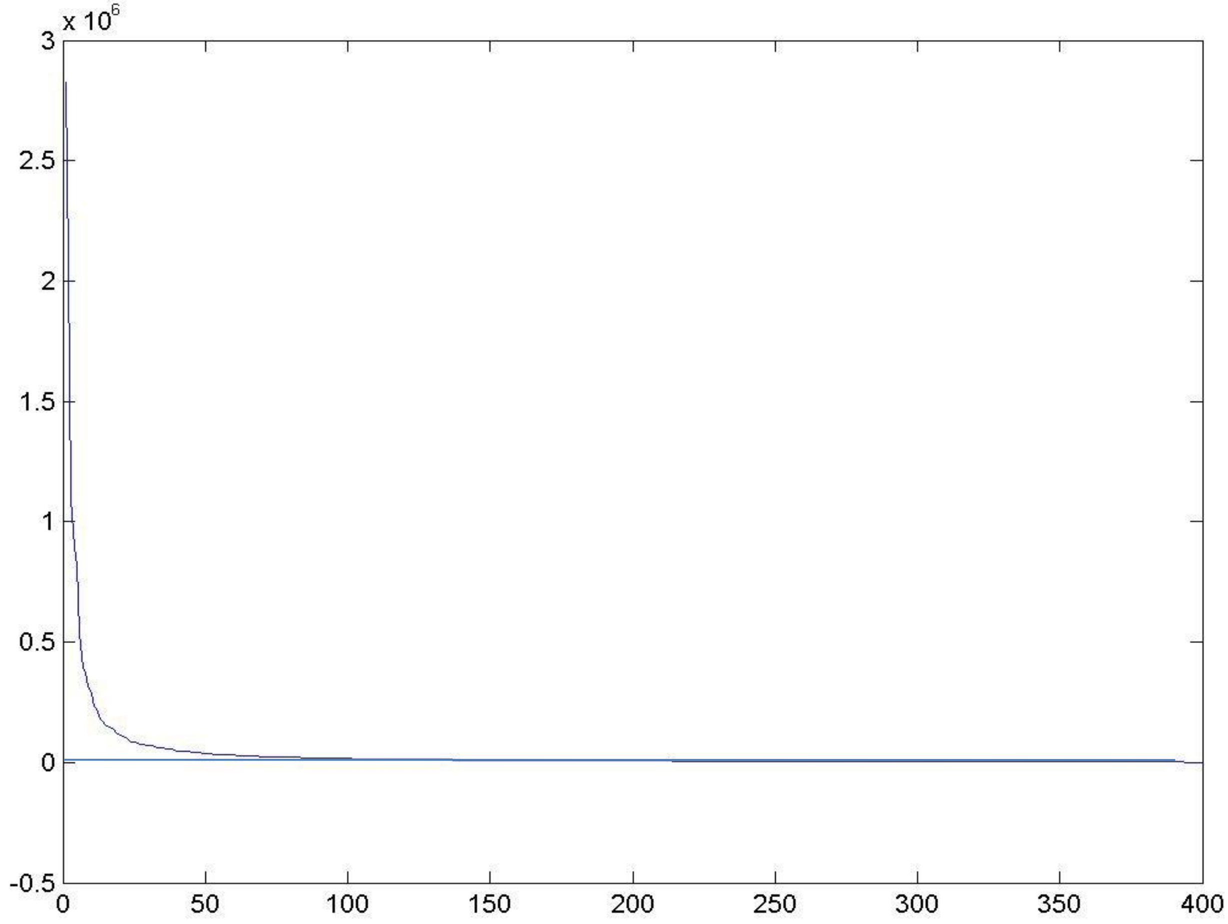
7. Visualize the reconstructed training face x_i

$$x_i \approx \mu + a_1\varphi_1 + a_2\varphi_2 + \ldots + a_K\varphi_K$$

Visualization of eigenfaces



Eigenvalues (variance along eigenvectors)



Reconstruction and Errors

K = 4

K = 200

K = 400

- Only selecting the top K eigenfaces, reduces the dimensionality.
- Fewer eigenfaces result in more information loss, and hence less discrimination between faces.

Eigenface algorithm

- Testing
  1. Take query image t
  2. Project into eigenface space and compute projection

$$t \rightarrow \left((t-\mu)\cdot\varphi_1, (t-\mu)\cdot\varphi_2, \ldots,\ (t-\mu)\cdot\varphi_K\right) \equiv \left(w_1, w_2, \ldots, w_K\right)$$

  3. Compare projection w with all N training projections
     - Simple comparison metric: Euclidean
     - Simple decision: K-Nearest Neighbor

       (note: this "K" refers to the k-NN algorithm, is different from the previous K's referring to the # of principal components)

Shortcomings

- Requires carefully controlled data:
  - All faces centered in frame
  - Same size
  - Some sensitivity to angle
- Alternative:
  - "Learn" one set of PCA vectors for each angle
  - Use the one with lowest error
- Method is completely knowledge free
  - (sometimes this is good!)
  - Doesn't know that faces are wrapped around 3D objects (heads)
  - Makes no effort to preserve class distinctions

Summary for Eigenface

Pros

- Non-iterative, globally optimal solution

Limitations

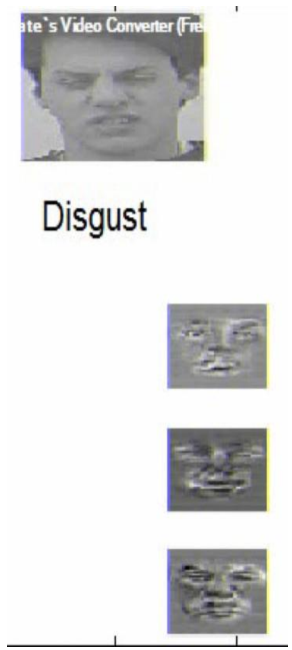
- PCA projection is **optimal for reconstruction** from a low dimensional basis, but **may NOT be optimal for discrimination…** Is there a better dimensionality reduction?

Besides face recognitions, we can also do Facial expression recognition

Happiness subspace (method A)

Disgust subspace (method A)

Facial Expression Recognition Movies (method A)



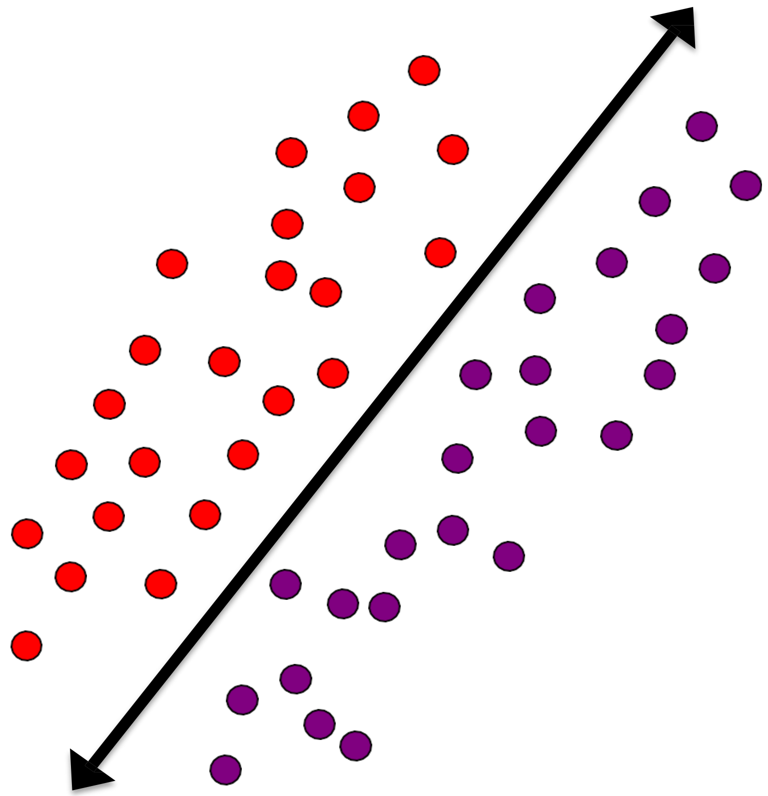
What we will learn today

- Feature normalization
- Introduction to face recognition
- A simple recognition pipeline with kNN
- The Eigenfaces Algorithm
- **Linear Discriminant Analysis (LDA)**

Turk and Pentland, Eigenfaces for Recognition, *Journal of Cognitive Neuroscience* **3** (1): 71–86, 1991.

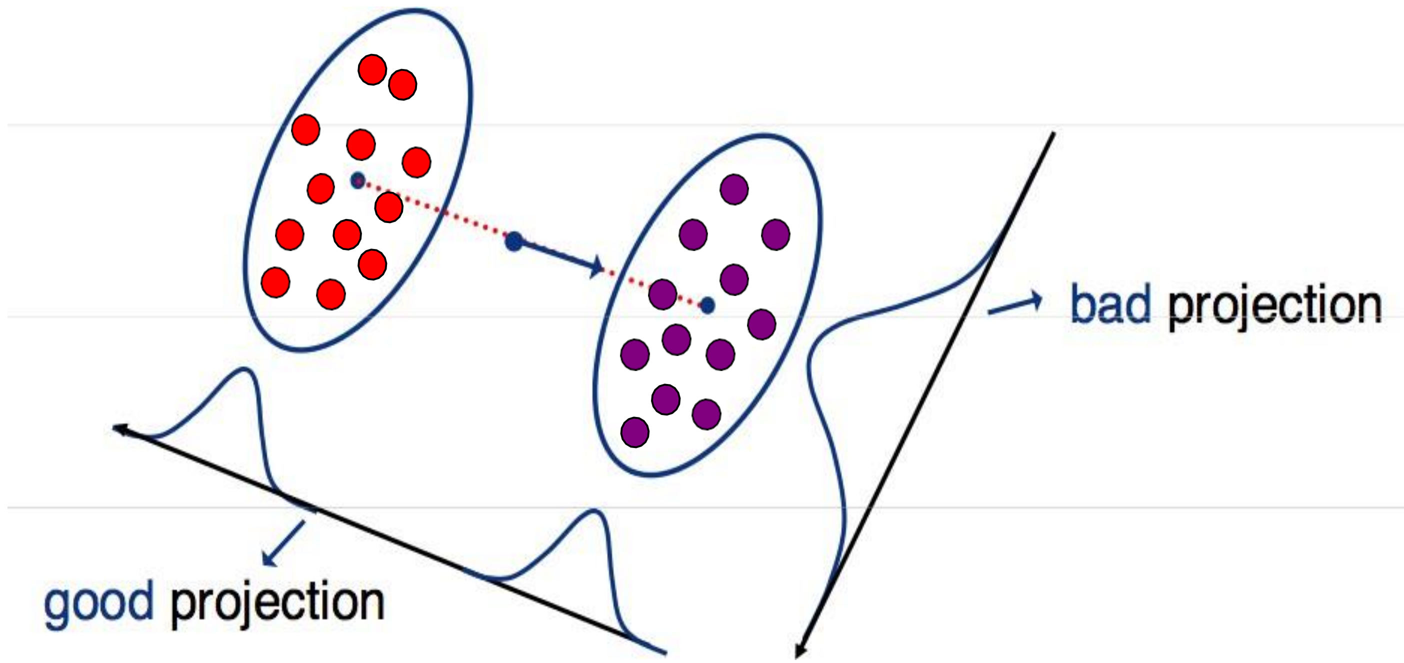
P. Belhumeur, J. Hespanha, and D. Kriegman. "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". *IEEE Transactions on pattern analysis and machine intelligence* **19** (7): 711. 1997.

Which direction will be the first principle component?



Fischer’s Linear Discriminant Analysis

- Goal: find the best separation between two classes



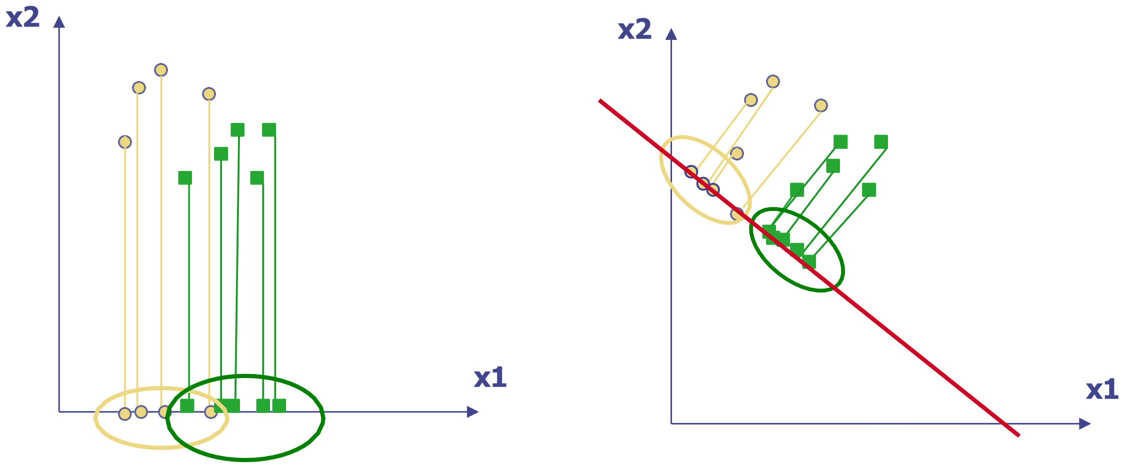
Slide inspired by N. Vasconcelos

Difference between PCA and LDA

- PCA preserves maximum variance
- LDA preserves discrimination
  - Find projection that maximizes scatter between classes and minimizes scatter within classes

Illustration of the Projection

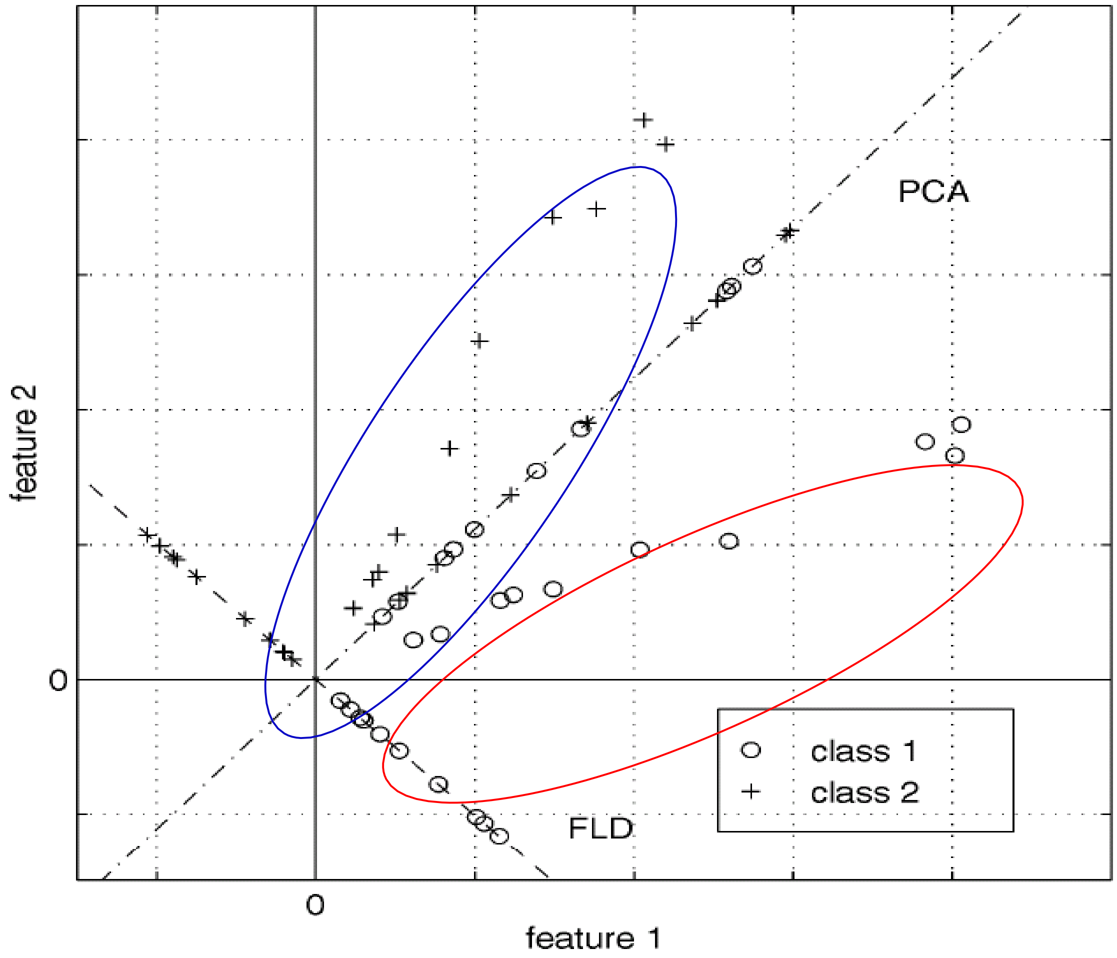
- Using two classes as example:



Poor Projection

Good

Basic intuition: PCA vs. LDA



LDA with 2 variables

- We want to learn a projection W such that the projection converts all the points from x to a new space (For this example, assume m == 1):

$$z = w^{\mathrm{T}} x \qquad z \in \mathbf{R}^m \quad x \in \mathbf{R}^n$$

- Let the **per class** means be:

$$E_{X|Y}[X \mid Y = i] = \mu_i$$

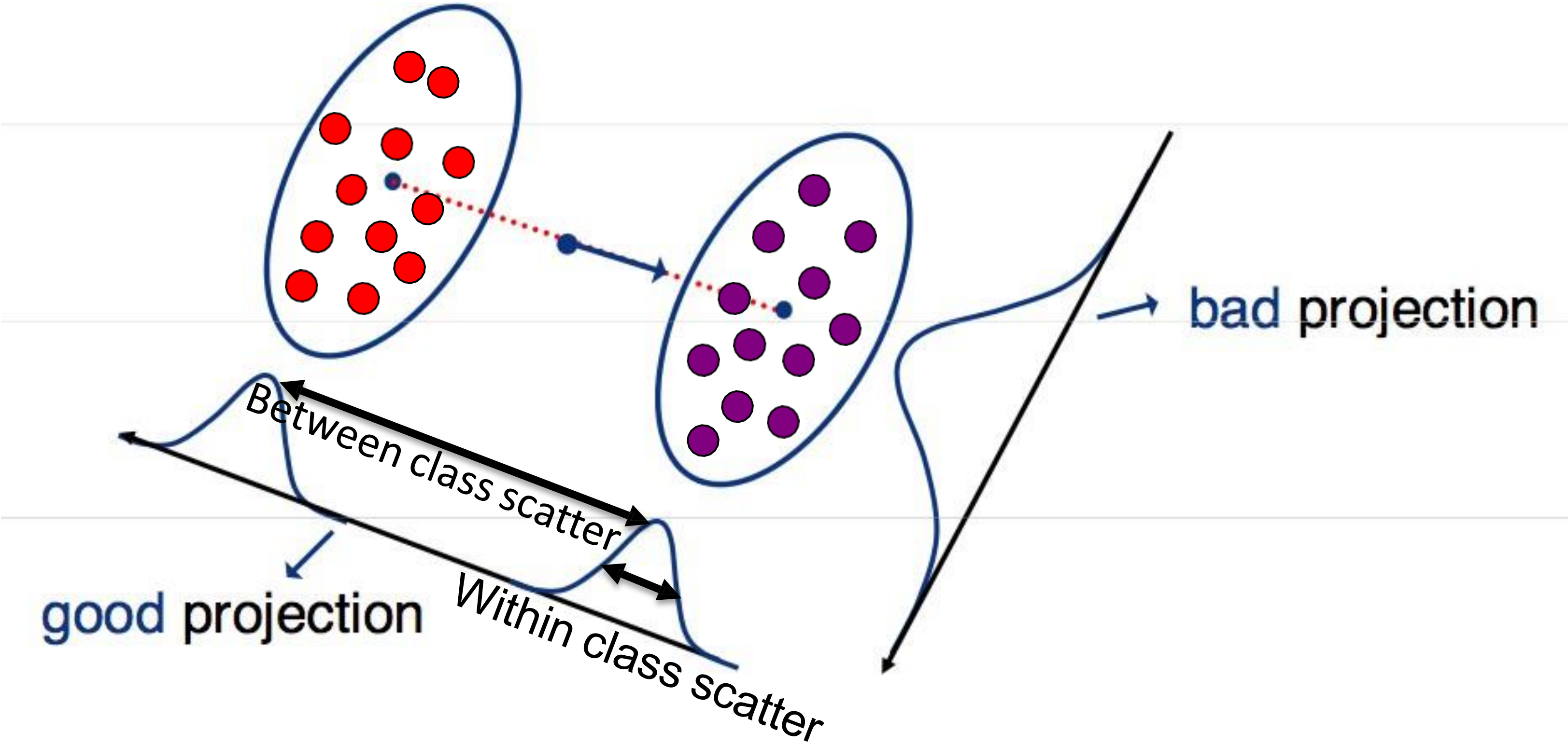
- And the **per class** covariance matrices be:

$$[(X - \mu_i)(X - \mu_i)^T \mid Y = i] = \Sigma_i$$

- We want a projection that maximizes:

$$J(w) = \max \frac{between\ \ class\ \ scatter}{within\ \ class\ \ scatter}$$

Fischer's Linear Discriminant Analysis



Slide inspired by N. Vasconcelos

LDA with 2 variables

The following objective function:

$$J(w) = \max \frac{between\ \ class\ \ scatter}{within\ \ class\ \ scatter}$$

Can be written as

$$J(w) = \frac{\left(E_{Z|Y}[Z \mid Y=1] - E_{Z|Y}[Z \mid Y=0]\right)^2}{\mathrm{var}[Z \mid Y=1] + \mathrm{var}[Z \mid Y=0]}$$

LDA with 2 variables

- We can write the between class scatter as:

$$\left(E_{Z|Y}[Z \mid Y=1] - E_{Z|Y}[Z \mid Y=0]\right)^2 = \left(w^T[\mu_1 - \mu_0]\right)^2$$
$$= w^T[\mu_1 - \mu_0][\mu_1 - \mu_0]^T w$$

- Also, the within class scatter becomes:

$$\mathrm{var}[Z \mid Y=i] = E_{Z|Y}\left\{\left(z - E_{Z|Y}[Z \mid Y=i]\right)^2 \mid Y=i\right\}$$
$$= E_{Z|Y}\left\{\left(w^T[x-\mu_i]\right)^2 \mid Y=i\right\}$$
$$= E_{Z|Y}\left\{w^T[x-\mu_i][x-\mu_i]^T w \mid Y=i\right\}$$
$$= w^T \Sigma_i w$$

Slide inspired by N. Vasconcelos

LDA with 2 variables

- We can plug in these scatter values to our objective function:

between class scatter

$$J(w) = \frac{w^T S_B w}{w^T S_W w} \qquad \begin{aligned} S_B &= (\mu_1 - \mu_0)(\mu_1 - \mu_0)^T \\ S_W &= (\Sigma_1 + \Sigma_0) \end{aligned}$$

within class scatter

- And our objective becomes:

$$J(w) = \frac{\left(E_{Z|Y}[Z \mid Y=1] - E_{Z|Y}[Z \mid Y=0]\right)^2}{\mathrm{var}[Z \mid Y=1] + \mathrm{var}[Z \mid Y=0]}$$

$$= \frac{w^T(\mu_1 - \mu_0)(\mu_1 - \mu_0)^T w}{w^T(\Sigma_1 + \Sigma_0)w}$$

Slide inspired by N. Vasconcelos

LDA with 2 variables

- The scatter variables

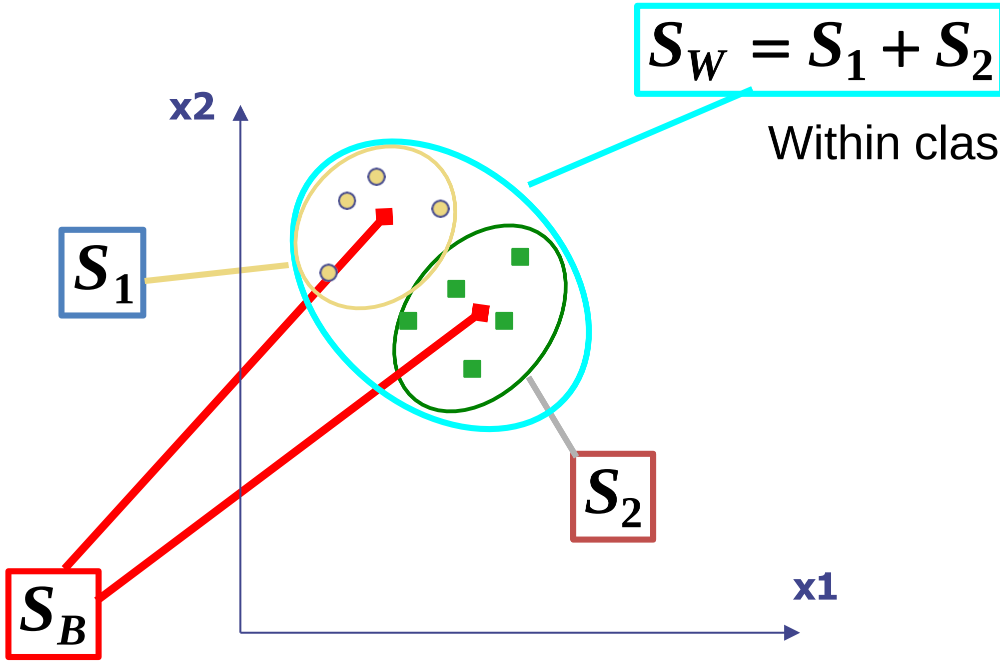
between class scatter

$$\boxed{\begin{aligned} S_B &= (\mu_1 - \mu_0)(\mu_1 - \mu_0)^T \\ S_W &= (\Sigma_1 + \Sigma_0) \end{aligned}}$$

within class scatter

Slide inspired by N. Vasconcelos

Visualization



Between class scatter

Linear Discriminant Analysis (LDA)

- Maximizing the ratio

$$J(w) = \frac{w^T S_B w}{w^T S_W w}$$

- Is equivalent to maximizing the numerator while keeping the denominator constant, i.e.

$$\boxed{\max_w w^T S_B w \quad \text{subject to} \quad w^T S_W w = K}$$

- And can be accomplished using Lagrange multipliers, where we define the Lagrangian as

$$\boxed{L = w^T S_B w - \lambda\left(w^T S_W w - K\right)}$$

- And maximize with respect to both w and λ

Slide inspired by N. Vasconcelos

Linear Discriminant Analysis (LDA)

- Setting the gradient of

$$L = w^T\left(S_B - \lambda S_W\right)w + \lambda K$$

With respect to w to zeros we get

$$\nabla_w L = 2\left(S_B - \lambda S_W\right)w = 0$$

or

$$\boxed{S_B w = \lambda S_W w}$$

- This is a generalized eigenvalue problem
- The solution is easy when $\boxed{S_w^{-1} = \left(\Sigma_1 + \Sigma_0\right)^{-1}}$

Slide inspired by N. Vasconcelos

Linear Discriminant Analysis (LDA)

- In this case

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

- And using the definition of S_B

$$S_W^{-1} (\mu_1 - \mu_0)(\mu_1 - \mu_0)^T w = \lambda w$$

- Assuming that $(\mu_1 - \mu_0)^T w = \alpha$ is a scalar, this can be written as

$$S_W^{-1} (\mu_1 - \mu_0) = \frac{\lambda}{\alpha} w$$

- and since we don't care about the magnitude of w

$$\boxed{w^* = S_W^{-1} (\mu_1 - \mu_0) = (\Sigma_1 + \Sigma_0)^{-1} (\mu_1 - \mu_0)}$$

Slide inspired by N. Vasconcelos

LDA with n variables and C classes

Variables

- N Sample images: $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}$
- C classes: $\{Y_1, Y_2, \ldots, Y_c\}$
- Average of each class: $\boldsymbol{\mu}_i = \frac{1}{N_i} \sum_{x_k \in Y_i} \boldsymbol{x}_k$
- Average of all data: $\boldsymbol{\mu} = \frac{1}{N} \sum_{k=1}^{N} \boldsymbol{x}_k$

Scatter Matrices

- Scatter of class i: $S_i = \sum_{x_k \in Y_i} (x_k - \mu_i)(x_k - \mu_i)^T$
- Within class scatter: $S_W = \sum_{i=1}^{c} S_i$
- Between class scatter: $S_B = \sum_{i=1}^{C} \sum_{j \neq i} (\mu_i - \mu_j)(\mu_i - \mu_j)^T$ Time-consuming to compute
- Between class scatter (in practice):

$$S_B = S_T - S_W = \sum_{i=1}^{C} N_i (\mu_i - \mu)(\mu_i - \mu)^T$$

- Total scatter: $S_T = \sum_{\boldsymbol{x}} (\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^T$

Mathematical Formulation

- Recall that we want to learn a projection W such that the projection converts all the points from x to a new space z:

$$z = w^{\mathrm{T}} x \qquad z \in \mathbf{R}^m \quad x \in \mathbf{R}^n$$

- After projection:
  - Between class scatter $\tilde{S}_B = W^T S_B W$
  - Within class scatter $\tilde{S}_W = W^T S_W W$
- So, the objective becomes:

$$W_{opt} = \arg\max_{W} \frac{|\tilde{S}_B|}{|\tilde{S}_W|} = \arg\max_{W} \frac{|W^T S_B W|}{|W^T S_W W|}$$

Mathematical Formulation

$$W_{opt} = \arg\max_{W} \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|}$$

- Solve generalized eigenvector problem:

$$S_B w_i = \lambda_i S_W w_i \qquad i = 1, \ldots, m$$

Mathematical Formulation

- Solution: Generalized Eigenvectors

$$S_B w_i = \lambda_i S_W w_i \qquad\qquad i = 1, \ldots, m$$

- Rank of W_{opt} is limited
  - Rank(S_B) <= C-1
  - Rank(S_W) <= N-C

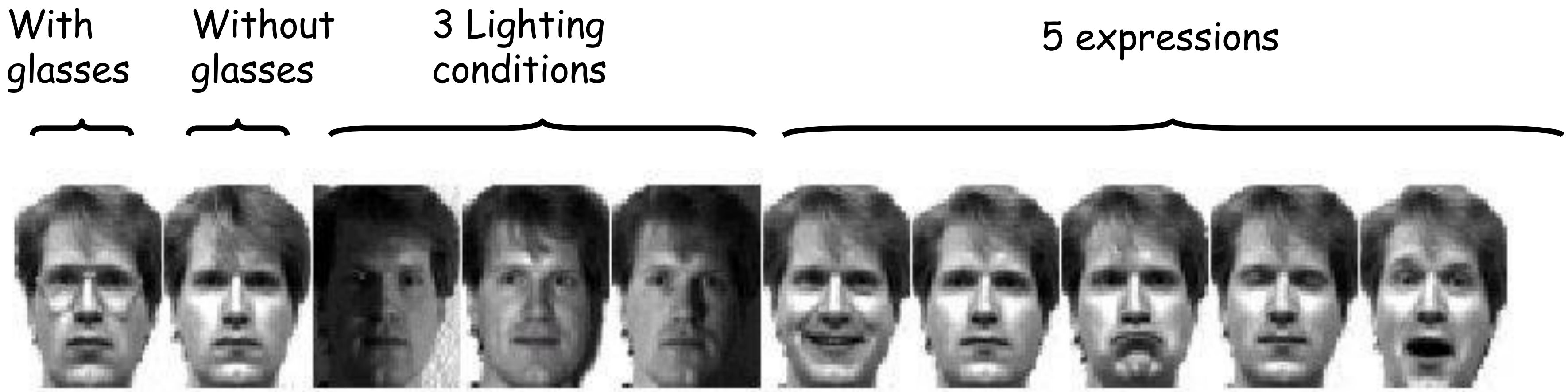
PCA vs. LDA

- Eigenfaces exploit the max scatter of the training images in face space
- Fisherfaces attempt to maximise the **between class scatter**, while minimising the **within class scatter**.

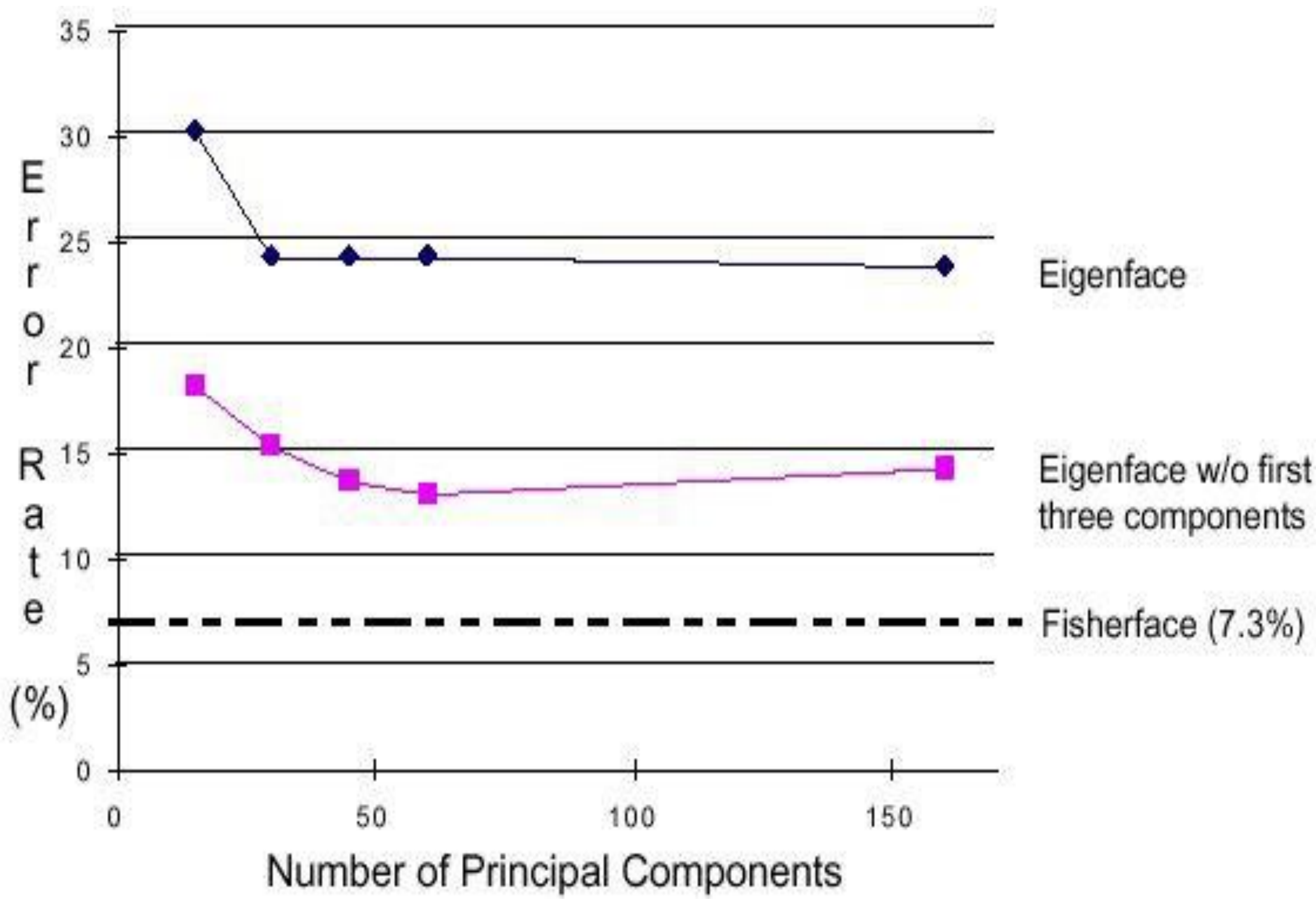
Results: Eigenface vs. Fisherface

- Input: 160 images of 16 people
- Train: 159 images
- Test: 1 image

- Variation in Facial Expression, Eyewear, and Lighting



Eigenface vs. Fisherface



What we have learned today

- Feature normalization
- Introduction to face recognition
- The Eigenfaces Algorithm
- Linear Discriminant Analysis (LDA)

Turk and Pentland, Eigenfaces for Recognition, *Journal of Cognitive Neuroscience* **3** (1): 71–86.

P. Belhumeur, J. Hespanha, and D. Kriegman. "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". *IEEE Transactions on pattern analysis and machine intelligence* **19** (7): 711. 1997.