Recognition, Detection, and Classification (Part 2)

Introduction to Computer Vision I

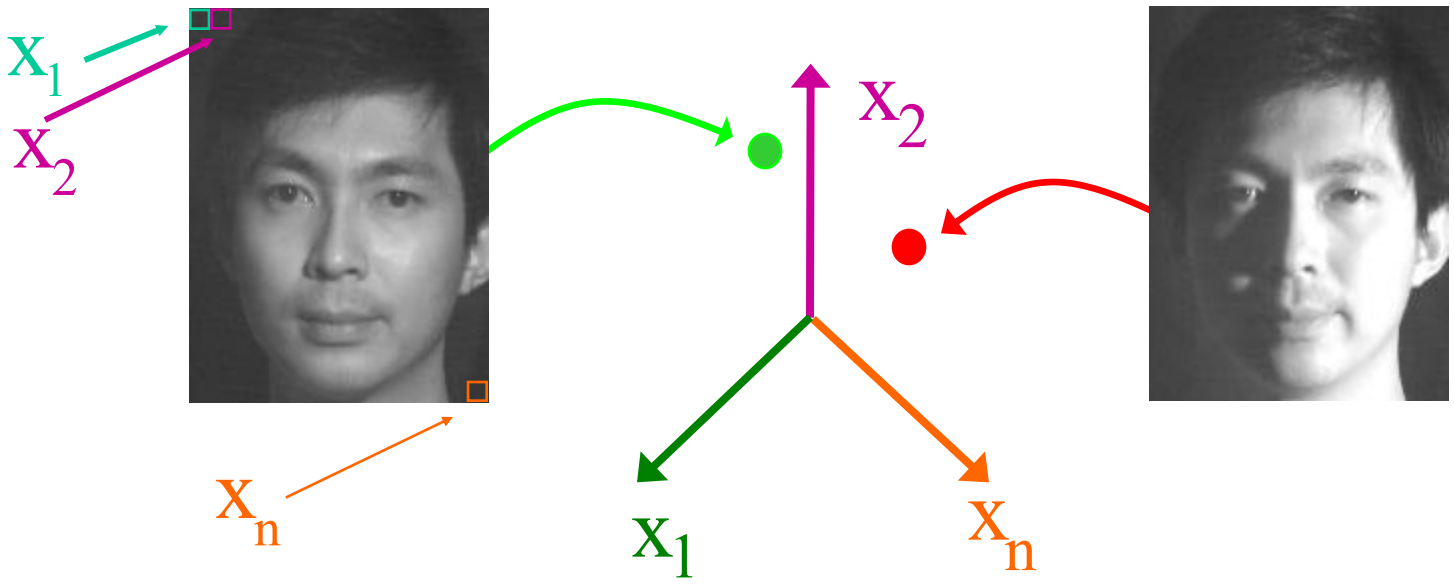
CSE 152A

Lecture 14

Announcements

- Assignment 3 is due Nov 20, 11:59 PM
- Assignment 4 will be released today
  - Due Dec 2, 11:59 PM

Simplest feature: Image as a Feature Vector



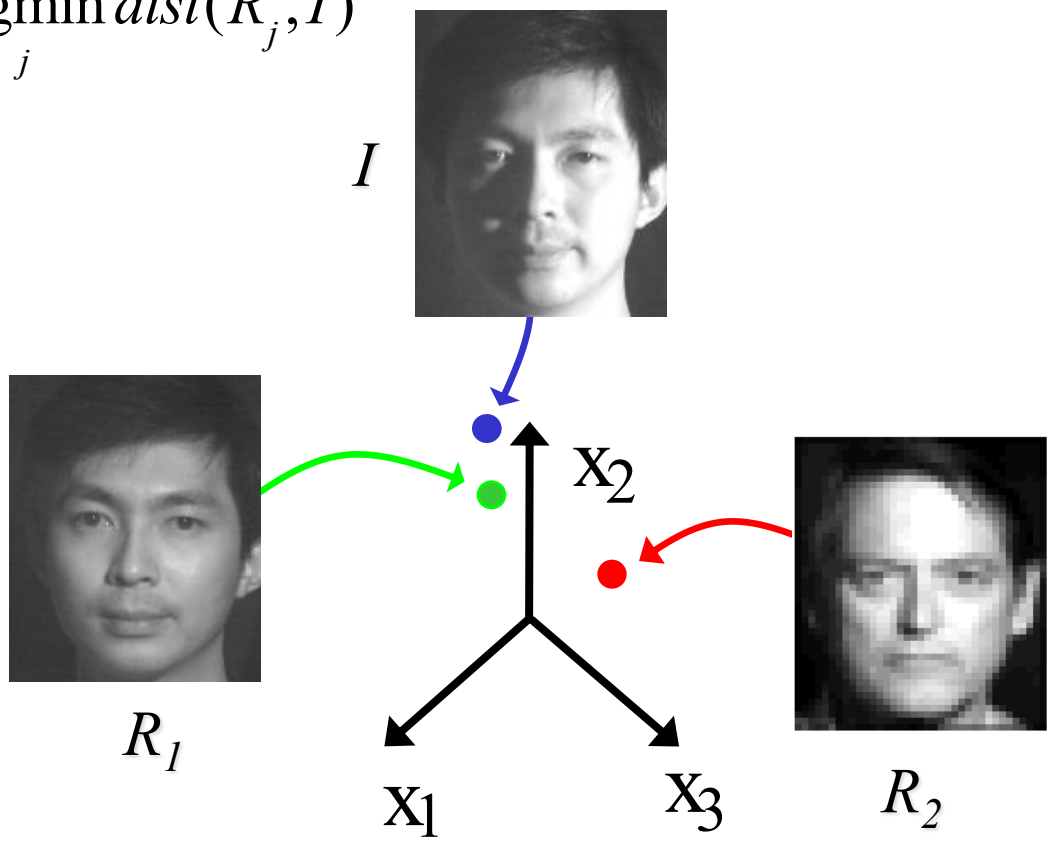
- Consider an n-pixel image to be a point in an n-dimensional space, $\mathbf{x} \in \mathbf{R}^n$
- Each pixel value $\mathbf{x_i}$ is a coordinate of $\mathbf{x}$

Nearest Neighbor Classifier

n classes (different people)

$\{R_j : j=1,\ldots, n\}$: set of training images, one per person

$$label = \underset{j}{\text{argmin}}\, dist(R_j, I)$$



k Nearest Neighbor Classification

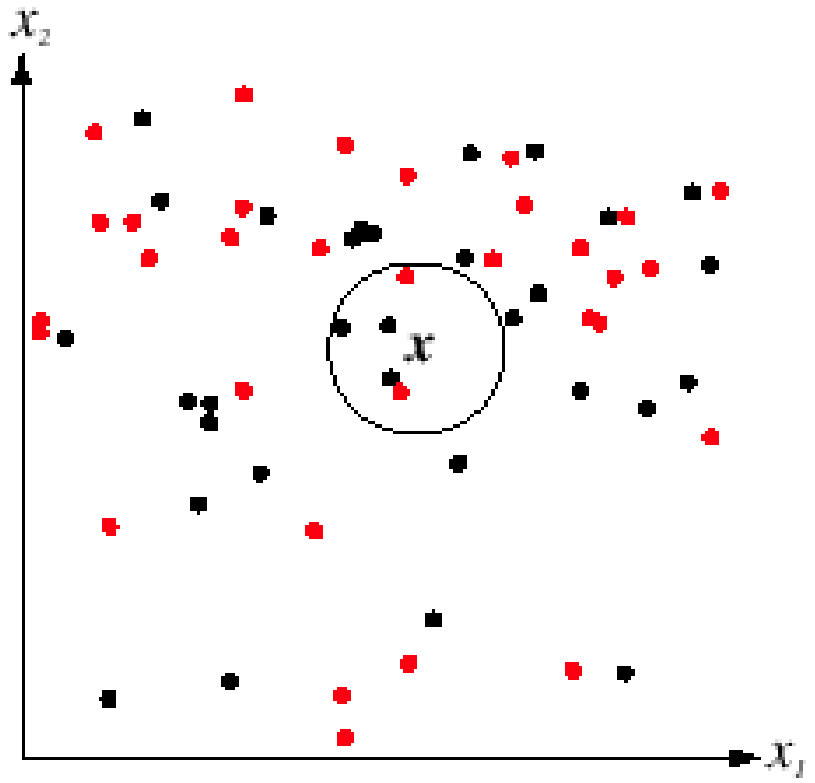
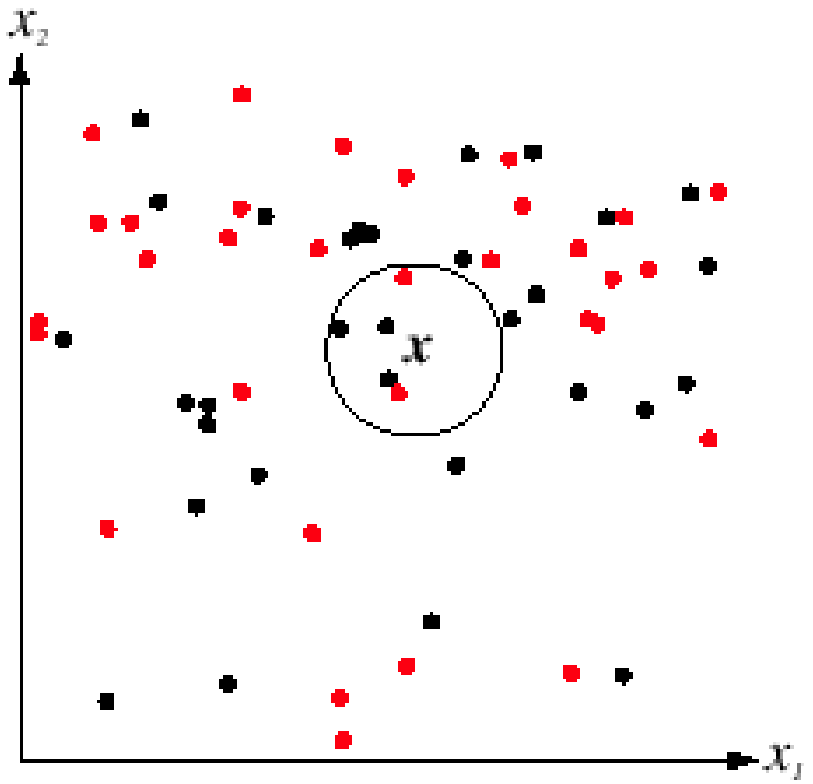


FIGURE 4.15. The k-nearest-neighbor query starts at the test point $\mathbf{x}$ and grows a spherical region until it encloses k training samples, and it labels the test point by a majority vote of these samples. In this $k = 5$ case, the test point $\mathbf{x}$ would be labeled the category of the black points. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Nearest Neighbor Classification



- k Nearest Neighbor with k=1
- It can be shown that as number of samples approaches infinity, the error rate of nearest neighbor is at most twice that of the optimal Bayesian classifier. (Why is even close? Training samples are drawn from $P(x \mid \omega_i)P(\omega_i)$)

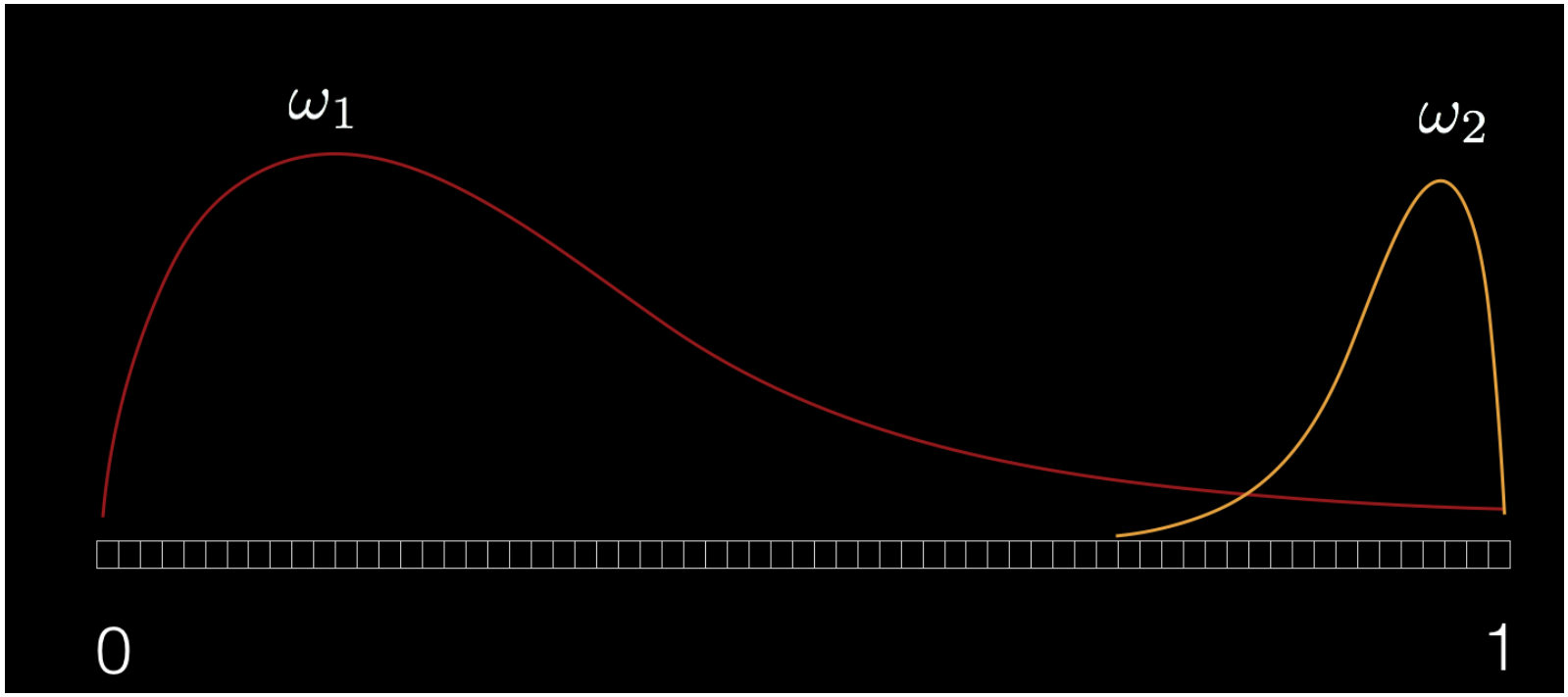
Comments on Nearest Neighbor

- Sometimes called "Template Matching"
- Variations on distance function (e.g., L_1, robust distances)
- Multiple templates per class- perhaps many training images per class
- Expensive to compute k distances, especially when each image is big (N dimensional)
- May not generalize well to unseen examples of class
- Will it apply well to other features (height, weight?)
- Some solutions:
  - Bayesian classification
  - Dimensionality reduction

- If we want to build a minimum-error rate classifier, then we need a very good estimate of $P(\omega_i | \mathbf{x})$

- How do we do this?

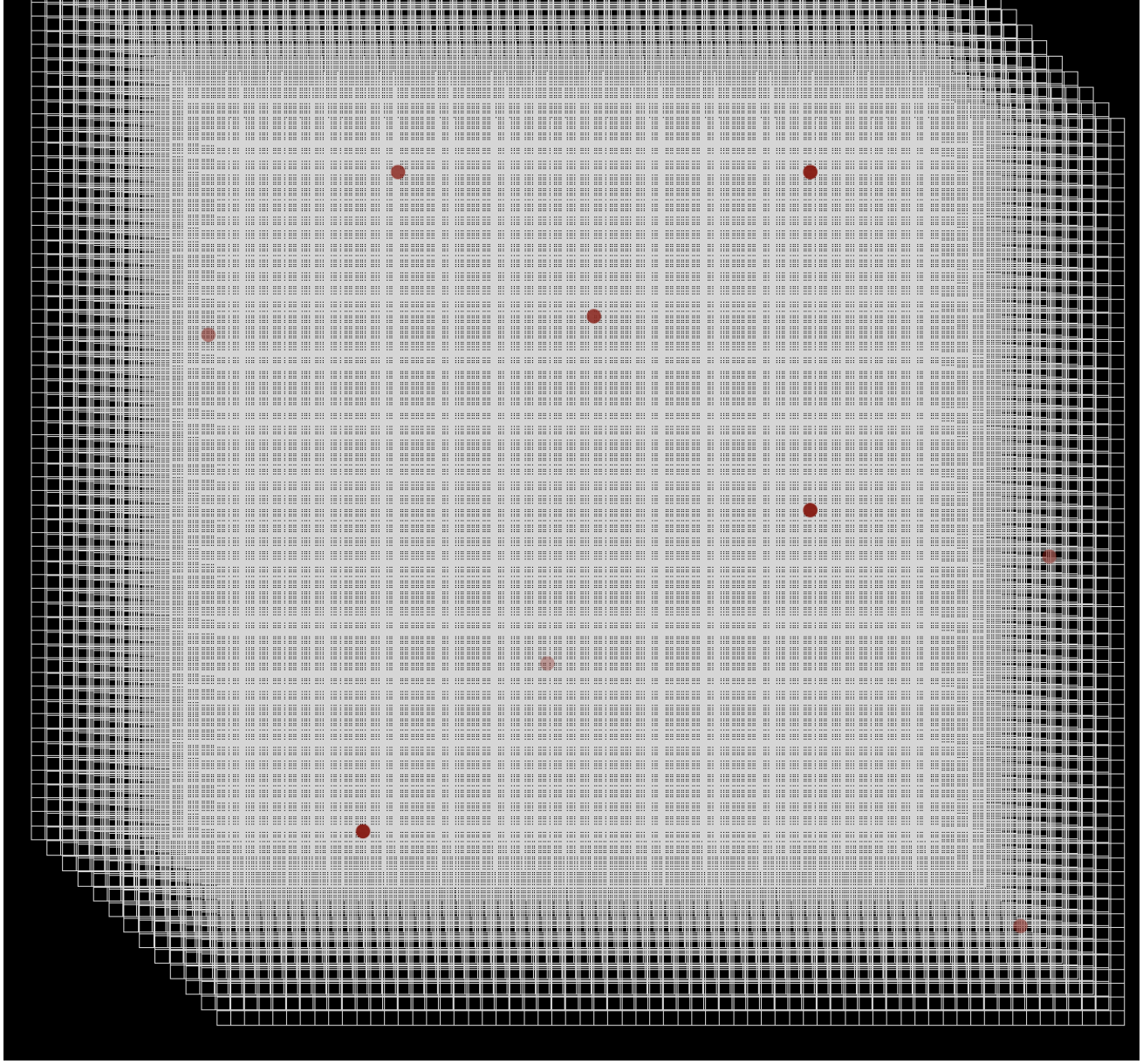
- Let's say our feature space is just 1-dimensional and our feature $\mathbf{x} \in [0, 1]$
- Let's say we have **10,000 training samples** from which to estimate our *a posteriori* probabilities
- We could estimate these probabilities using a histogram in which we divided the interval into 100 evenly spaced bins



- On average each bin would have 100 samples
- We could estimate $P(\mathbf{x} \mid \omega_i)$ as the number of samples from class i that fall in the same bin that falls into divided by the total number of samples in that bin

But this plan does not scale as we increase the dimensionality of the feature space!

- Let's say our feature space is 3 dimensional and our feature $\mathbf{x} \in [0, 1]^3$
- Let's say we still have **10,000 training samples** from which to estimate our *a posteriori* probabilities
- If we estimate these probabilities using a histogram in which we divide the volume into the same width bins as before…



On average each bin would only have 0.01 samples!

We're not going be able to estimate probabilities well

Curse of Dimensionality

Dimensionality Reduction

An idea:

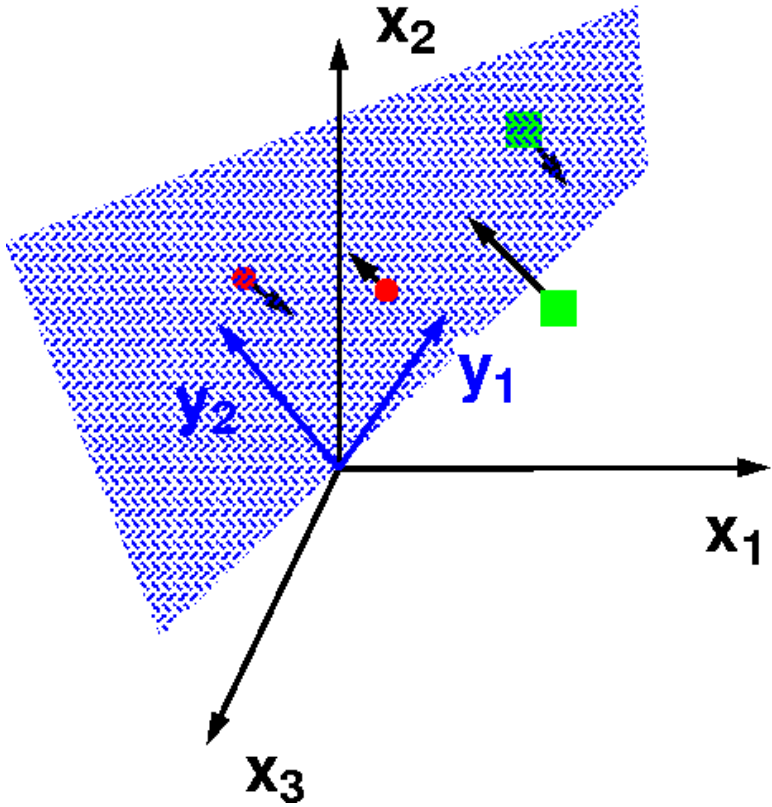
Represent the set of images as a linear subspace

What is a linear subspace?

Let V be a vector space and let W be a subset of V. Then W is a subspace if and only if:

1. The null vector $\mathbf{0}$ is in W
2. If $\mathbf{u}$ and $\mathbf{v}$ are elements of W, then any linear combination of $\mathbf{u}$ and $\mathbf{v}$ is an element of W; $a\mathbf{u} + b\mathbf{v} \in W$
3. If $\mathbf{u}$ is an element of W and c is a scalar, then the scalar product $c\mathbf{u} \in W$

- A k-dimensional subspace is spanned by k linearly independent vectors. It is spanned by a k-dimensional orthogonal basis



Example: A 2-D linear subspace of $\mathbf{R}^3$ spanned by y_1 and y_2

Linear Subspaces & Linear Projection

• A d-pixel image $\boldsymbol{x} \in \mathbf{R}^d$ can be projected to a low-dimensional feature space $\boldsymbol{y} \in \mathbf{R}^k$ by

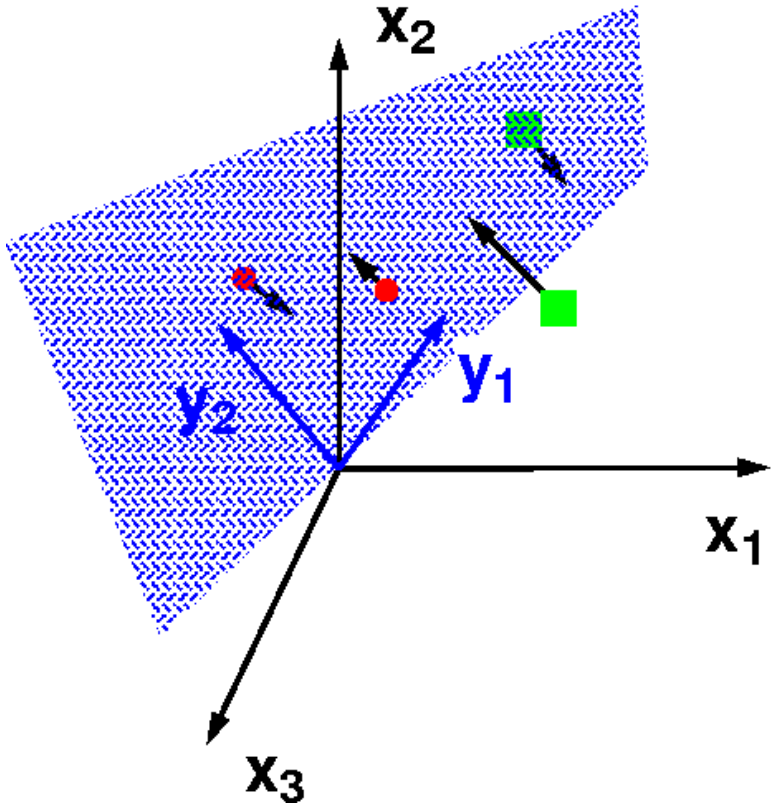
$$\boldsymbol{y} = W\boldsymbol{x}$$

where W is an k by d matrix

• Each training image is projected to the subspace

• Recognition is performed in $\mathbf{R}^k$ using, for example, nearest neighbor

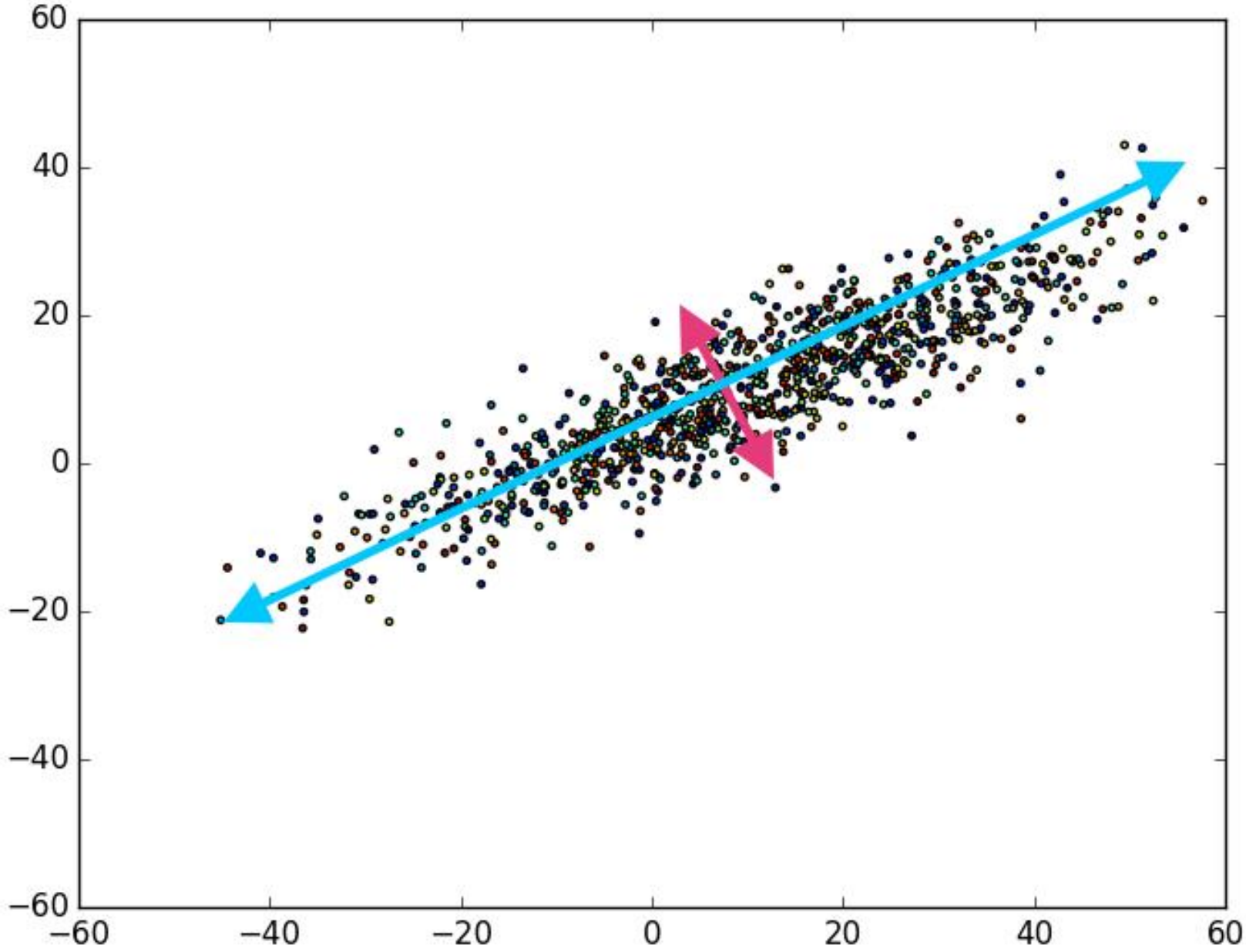
• How do we choose a good W?



Example: A 2-D linear subspace of $\mathbf{R}^3$ spanned by y_1 and y_2

How do we choose a good W?

- Drop dimensions (feature selection)
- Random projections
- Principal Component Analysis
- Linear Discriminant Analysis
- Independent Component Analysis
- Or non-linear dimensionality reduction


60
40
20
0
−20
−40
−60
−60
−40
−20
0
20
40
60

Principal component analysis (PCA) of covariance matrix

Assume we have a set of m feature vectors $\mathbf{x}_i$ $(i = 1, \ldots, m) \in \mathbb{R}^n$. Write

$$\text{Mean } \boldsymbol{\mu}_{\mathbf{x}} = \frac{1}{m} \sum_i \mathbf{x}_i$$

$$\text{Covariance } \Sigma_{\mathbf{x}} = \frac{1}{m-1} \sum_i (\mathbf{x}_i - \boldsymbol{\mu}_{\mathbf{x}})(\mathbf{x}_i - \boldsymbol{\mu}_{\mathbf{x}})^\top$$

Eigen decomposition of covariance matrix

$$\Sigma_{\mathbf{x}} = \mathtt{V} \Lambda \mathtt{V}^\top$$

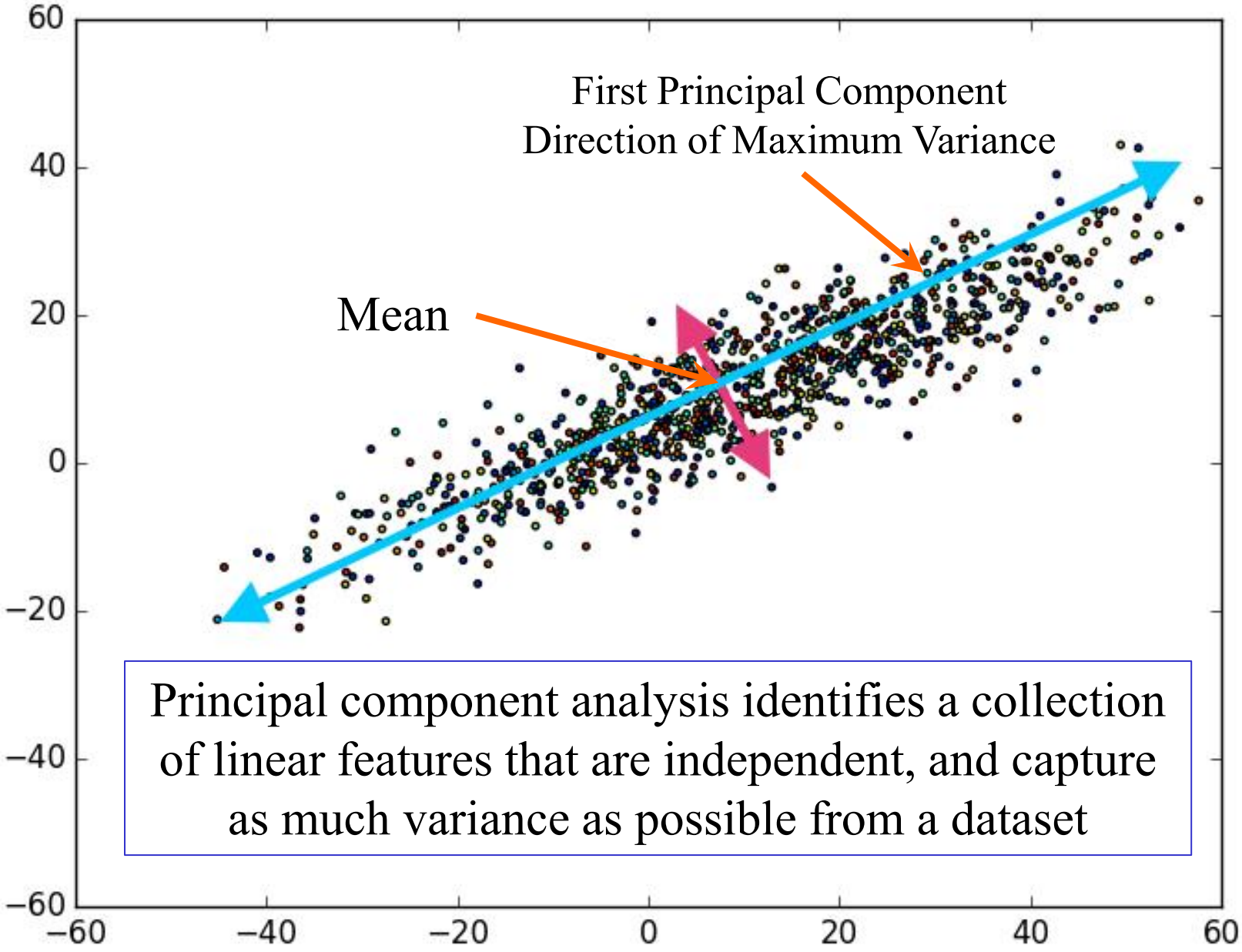
where

$\Sigma_{\mathbf{x}}$ is a positive semidefinite $n \times n$ matrix

$\mathtt{V}$ is an $n \times n$ orthogonal matrix

$\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$, where $\lambda_i \geq \lambda_{i+1} \geq 0$

Columns of $\mathtt{V}$ are eigenvectors (also called principal component coefficients) corresponding to eigenvalues (also called principal component variances) $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)^\top$.



Principal component analysis identifies a collection of linear features that are independent, and capture as much variance as possible from a dataset

Relationship between singular value decomposition (SVD) and eigen decomposition

$$\mathtt{A} = \mathtt{U}\Sigma\mathtt{V}^\top$$
$$\mathtt{A}^\top\mathtt{A} = (\mathtt{U}\Sigma\mathtt{V}^\top)^\top\mathtt{U}\Sigma\mathtt{V}^\top$$
$$\mathtt{A}^\top\mathtt{A} = \mathtt{V}\Sigma^\top\mathtt{U}^\top\mathtt{U}\Sigma\mathtt{V}^\top$$
$$\mathtt{A}^\top\mathtt{A} = \mathtt{V}\Sigma^\top\Sigma\mathtt{V}^\top$$
$$\mathtt{A}^\top\mathtt{A} = \mathtt{V}\Lambda\mathtt{V}^\top, \text{ where } \Lambda = \Sigma^\top\Sigma$$

$$\mathtt{A} = \mathtt{U}\Sigma\mathtt{V}^\top$$
$$\mathtt{A}\mathtt{A}^\top = \mathtt{U}\Sigma\mathtt{V}^\top(\mathtt{U}\Sigma\mathtt{V}^\top)^\top$$
$$\mathtt{A}\mathtt{A}^\top = \mathtt{U}\Sigma\mathtt{V}^\top\mathtt{V}\Sigma^\top\mathtt{U}^\top$$
$$\mathtt{A}\mathtt{A}^\top = \mathtt{U}\Sigma\Sigma^\top\mathtt{U}^\top$$
$$\mathtt{A}\mathtt{A}^\top = \mathtt{U}\Lambda\mathtt{U}^\top, \text{ where } \Lambda = \Sigma\Sigma^\top$$

where

$\mathtt{U}$ and $\mathtt{V}$ are orthogonal matrices

$\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n)$, where $\sigma_i \geq \sigma_{i+1} \geq 0$

$\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$, where $\lambda_i \geq \lambda_{i+1} \geq 0$

variances $\lambda_i = \sigma_i^2 \, \forall \, i$

Data matrix

Data matrix

$$\mathtt{X} \in \mathbb{R}^{m \times n} \quad m \text{ observations, } n \text{ variables}$$

$$\mathtt{X} = \begin{bmatrix} \mathbf{x}^{1\top} \\ \mathbf{x}^{2\top} \\ \vdots \\ \mathbf{x}^{m\top} \end{bmatrix}$$

Data matrix, mean-deviation form

$$\hat{\mathtt{X}} = \begin{bmatrix} \mathbf{x}^{1\top} - \boldsymbol{\mu}_x^\top \\ \mathbf{x}^{2\top} - \boldsymbol{\mu}_x^\top \\ \vdots \\ \mathbf{x}^{m\top} - \boldsymbol{\mu}_x^\top \end{bmatrix}$$

$$\Sigma_{\mathbf{x}} = \frac{1}{m-1} \hat{\mathtt{X}}^\top \hat{\mathtt{X}}$$

Singular value decomposition of (mean-deviation form of) data matrix

$$\hat{\mathtt{X}} = \mathtt{U}\mathtt{D}\mathtt{V}^\top$$

Note: economy SVD can be used

where

$\mathtt{D} = \mathrm{diag}(\sigma_1, \ldots, \sigma_{\min(m,n)})$, where $\sigma_i \geq \sigma_{i+1} \geq 0$

columns of $\mathtt{V}$ (rows of $\mathtt{V}^\top$) are principal component coefficients of $\hat{\mathtt{X}}$

Projection of $\hat{\mathtt{X}}$ to principal component axes

$\hat{\mathtt{A}} = \hat{\mathtt{X}}\mathtt{V}$ (forward) projection to principal component scores

$\hat{\mathtt{A}}\mathtt{V}^\top = \hat{\mathtt{X}}$ back projection

Dimensionality reduction

Columns of $\mathtt{V}$ corresponding to smallest singular values can be removed

$\hat{\mathtt{A}}' = \hat{\mathtt{X}}\mathtt{V}'$ (forward) projection to principal component scores

$\hat{\mathtt{A}}'\mathtt{V}'^\top = \hat{\mathtt{X}}'$ back projection (with loss of data)

SVD Properties

- r = rank($\boldsymbol{A}$) = number of non-zero singular values
- $\boldsymbol{U}$, $\boldsymbol{V}$ give an orthonormal bases for the subspaces of $\boldsymbol{A}$:
  - 1st r columns of $\boldsymbol{U}$: Column space of $\boldsymbol{A}$
  - Last m - r columns of $\boldsymbol{U}$: Left nullspace of $\boldsymbol{A}$
  - 1st r columns of $\boldsymbol{V}$: Row space of $\boldsymbol{A}$
  - 1st n - r columns of $\boldsymbol{V}$: (Right) nullspace of $\boldsymbol{A}$
- *For some d where $d \leq r$, the first d column of U provide the best d-dimensional basis for columns of A in least squares sense.*

PCA for recognition

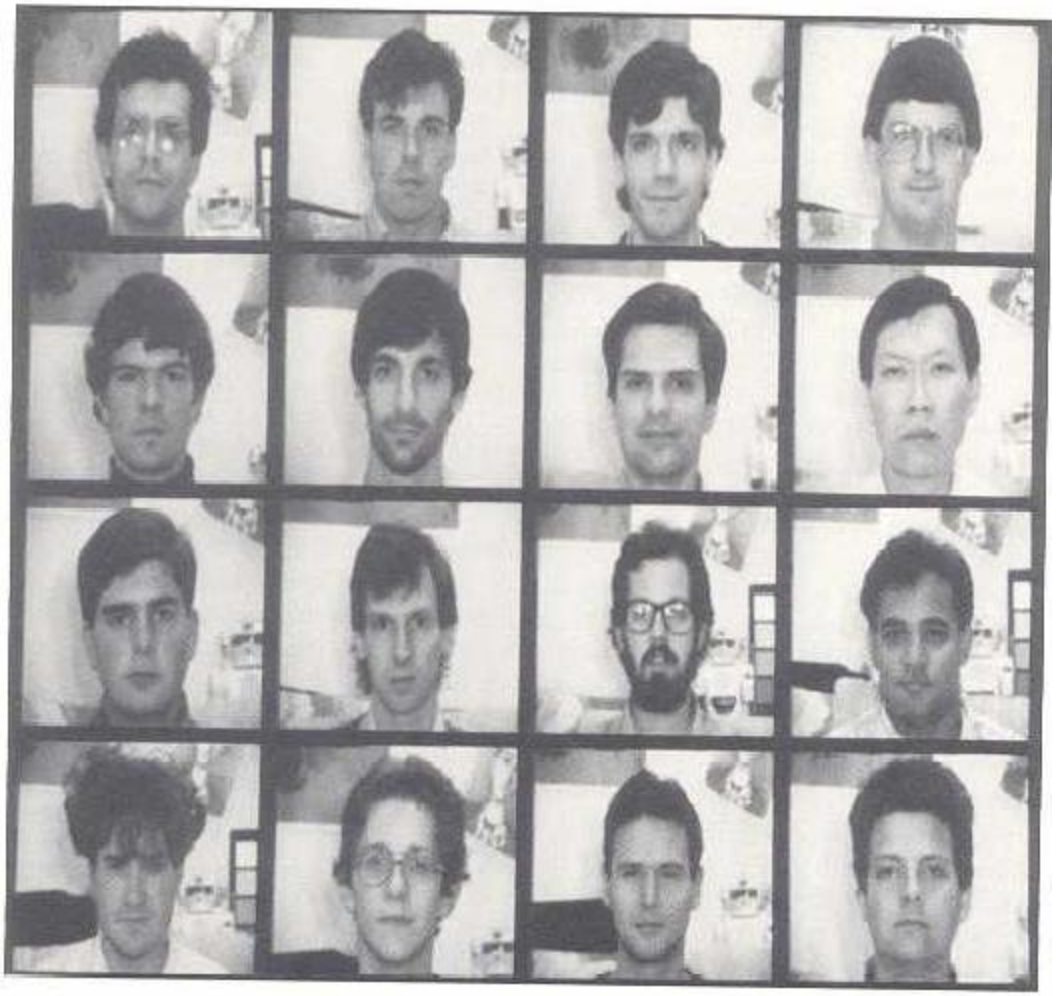
Modeling

1. Given a collection of n training images x_i, represent each one as a d-dimensional column vector
2. Compute the mean image and covariance matrix
3. Compute k Eigenvectors of the covariance matrix corresponding to the k largest Eigenvalues and form matrix $W^T=[u_1, u_2,...,u_k]$ (Or perform using SVD)
   - Note that the Eigenvectors are images
4. Project the training images to the k-dimensional Eigenspace. $y_i=Wx_i$

Recognition

1. Given a test image x, project the vectorized image to the Eigenspace by $y=Wx$
2. Perform classification of y to the projected training images

Example: Eigenfaces



Training images

[Turk, Pentland 91]

Eigenfaces



Mean Image



Basis Images

Accuracy of PCA + K-NN

KNN + 3,072 Features	33.86
KNN + 200 PCA Comp.	36.54
KNN + 75 PCA Comp.	39.77
KNN + 50 PCA Comp.	40.12
KNN + 40 PCA Comp.	40.93
KNN + 30 PCA Comp.	**41.78**
KNN + 25 PCA Comp.	41.57
KNN + 15 PCA Comp.	38.75
KNN + 10 PCA Comp.	34.93

Difficulties with PCA

- Projection may suppress important detail
  - smallest variance directions may not be unimportant
- Method does not take discriminative task into account
  - typically, we wish to compute features that allow good discrimination
  - not the same as largest variance or minimizing reconstruction error

Fisherfaces: Class specific linear projection

P. Belhumeur, J. Hespanha, D. Kriegman, *Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection*, PAMI, July 1997, pp. 711--720.

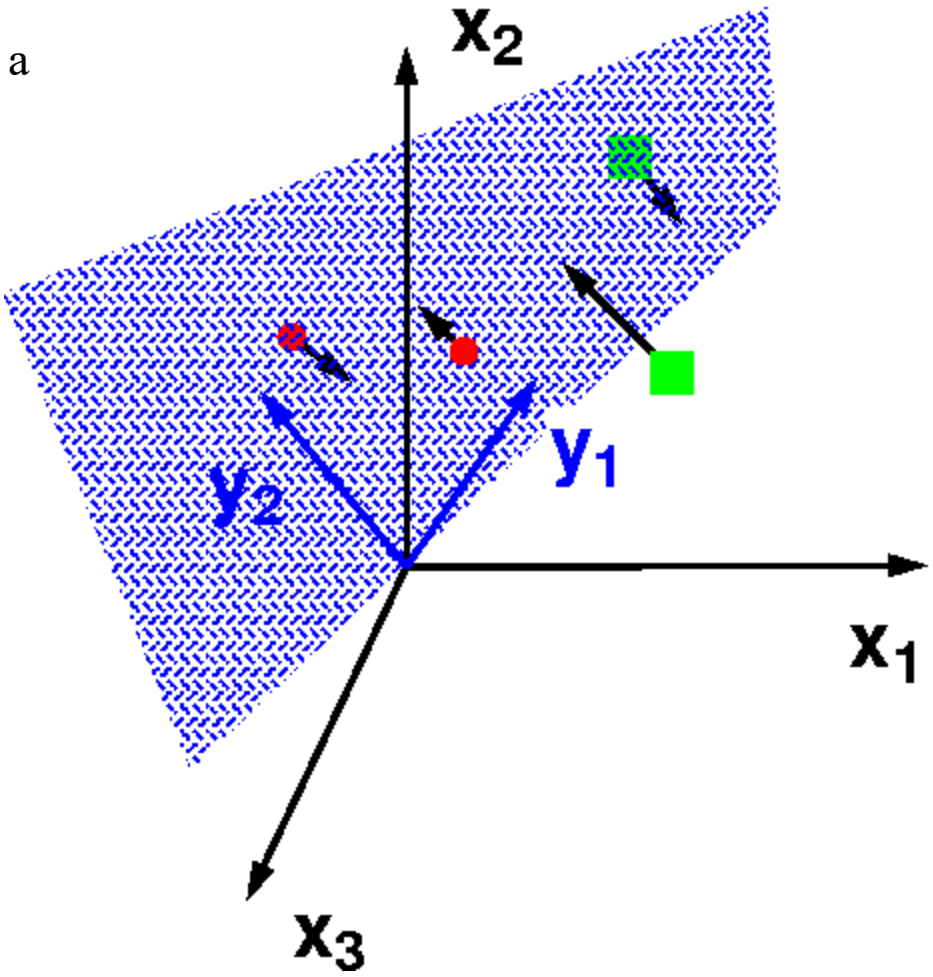
• An n-pixel image $\boldsymbol{x} \in \mathbf{R}^d$ can be projected to a low-dimensional feature space $\boldsymbol{y} \in \mathbf{R}^k$ by

$$\boldsymbol{y} = W\boldsymbol{x}$$

where W is an k by d matrix

• Recognition is performed using nearest neighbor in $\mathbf{R}^k$

• How do we choose a good W?



PCA & Fisher's Linear Discriminant

- Between-class scatter

$$S_B = \sum_{i=1}^{c} |\chi_i| (\mu_i - \mu)(\mu_i - \mu)^T$$

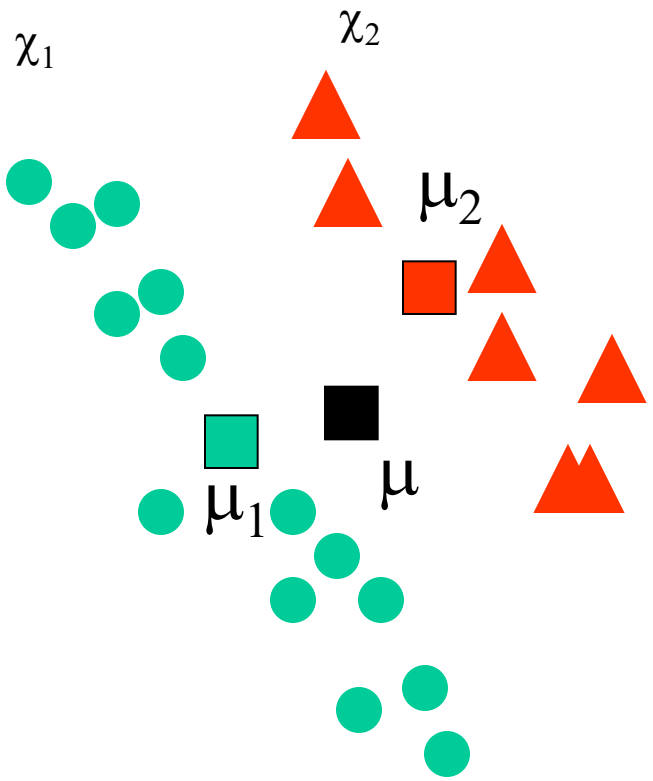
- Within-class scatter

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

- Total scatter

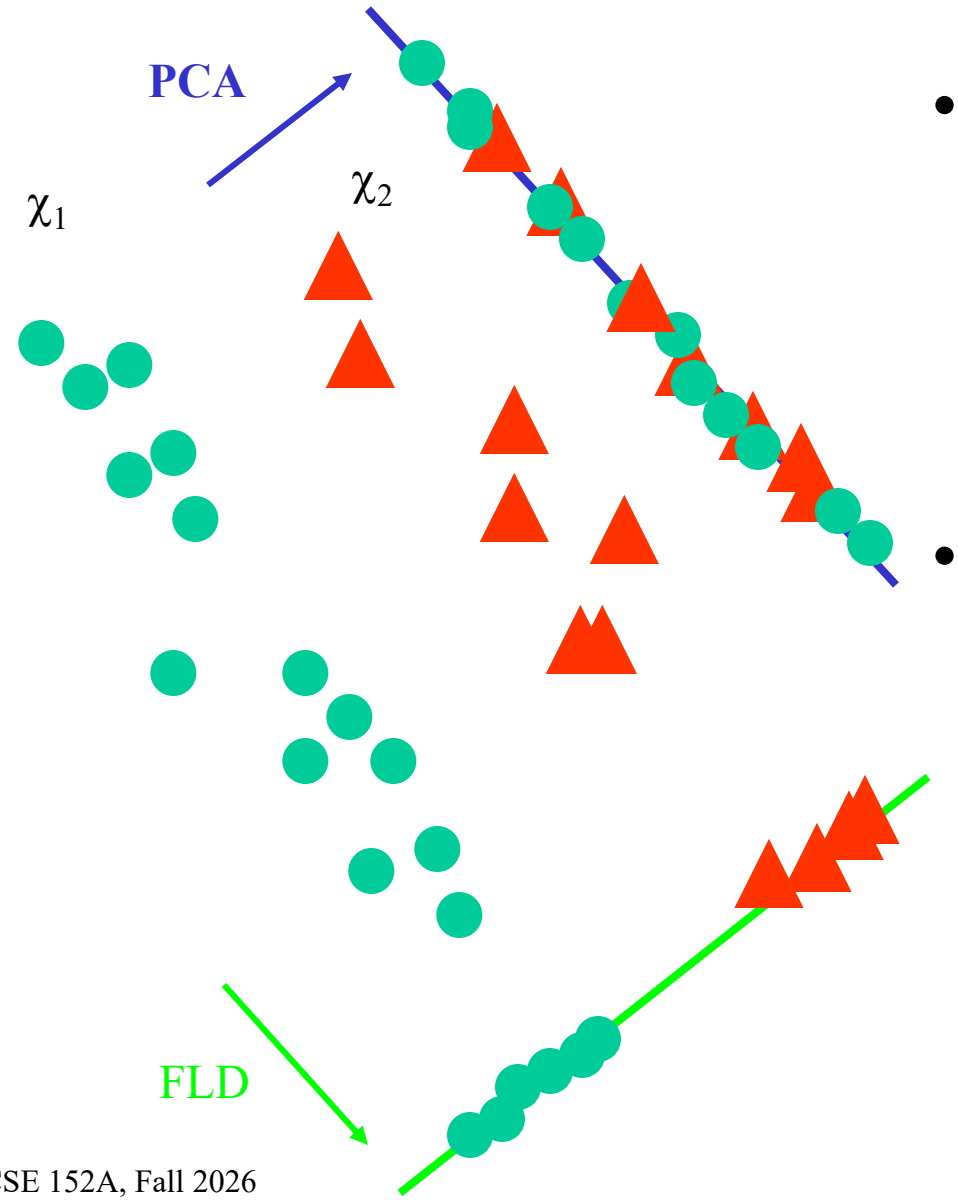
$$S_T = \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu)(x_k - \mu)^T = S_B + S_W$$

- Where
  - c is the number of classes
  - μ_i is the mean of class χ_i
  - $|\chi_i|$ is number of samples of χ_i.



If the data points x_i are projected by $y_i = Wx_i$ and the scatter of x_i is S, then the scatter of the projected points y_i is $W^T S W$

PCA & Fisher's Linear Discriminant



- PCA (Eigenfaces)

$$W_{PCA} = \arg\max_{W} \left| W^T S_T W \right|$$

Maximizes projected total scatter

- Fisher's Linear Discriminant

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

Maximizes ratio of projected between-class to projected within-class scatter

Computing the Fisher Projection Matrix

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

$$= \left[\mathbf{w}_1 \quad \mathbf{w}_2 \quad \ldots \quad \mathbf{w}_m\right] \qquad (4)$$

where $\left\{\mathbf{w}_i \,\middle|\, i = 1, 2, \ldots, m\right\}$ is the set of generalized eigenvectors of S_B and S_W corresponding to the m largest generalized eigenvalues $\left\{\lambda_i \,\middle|\, i = 1, 2, \ldots, m\right\}$, i.e.,

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \qquad i = 1, 2, \ldots, m$$

- There are at most c-1 non-zero generalized Eigenvalues, so m $\leq$ c-1

Generalized eigenvalues and generalized eigenvectors for a pair of symmetric matrices

$$\mathtt{AV} = \mathtt{BV}\Lambda \text{ and } \mathtt{V}^\top \mathtt{A} = \Lambda \mathtt{V}^\top \mathtt{B}$$

where

$\mathtt{A}$ is symmetric

$\mathtt{B}$ is (symmetric) positive definite

$\mathtt{V} = [\mathbf{v}_1 \mid \cdots \mid \mathbf{v}_n]$ is not orthogonal

$\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$

Eigenvalue λ_i corresponds to eigenvector $\mathbf{v}_i$.

$\mathtt{S}_\mathrm{B}$ is $\mathtt{A}$

$\mathtt{S}_\mathrm{W}$ is $\mathtt{B}$

Fisherfaces

$$W = W_{fld} W_{PCA}$$

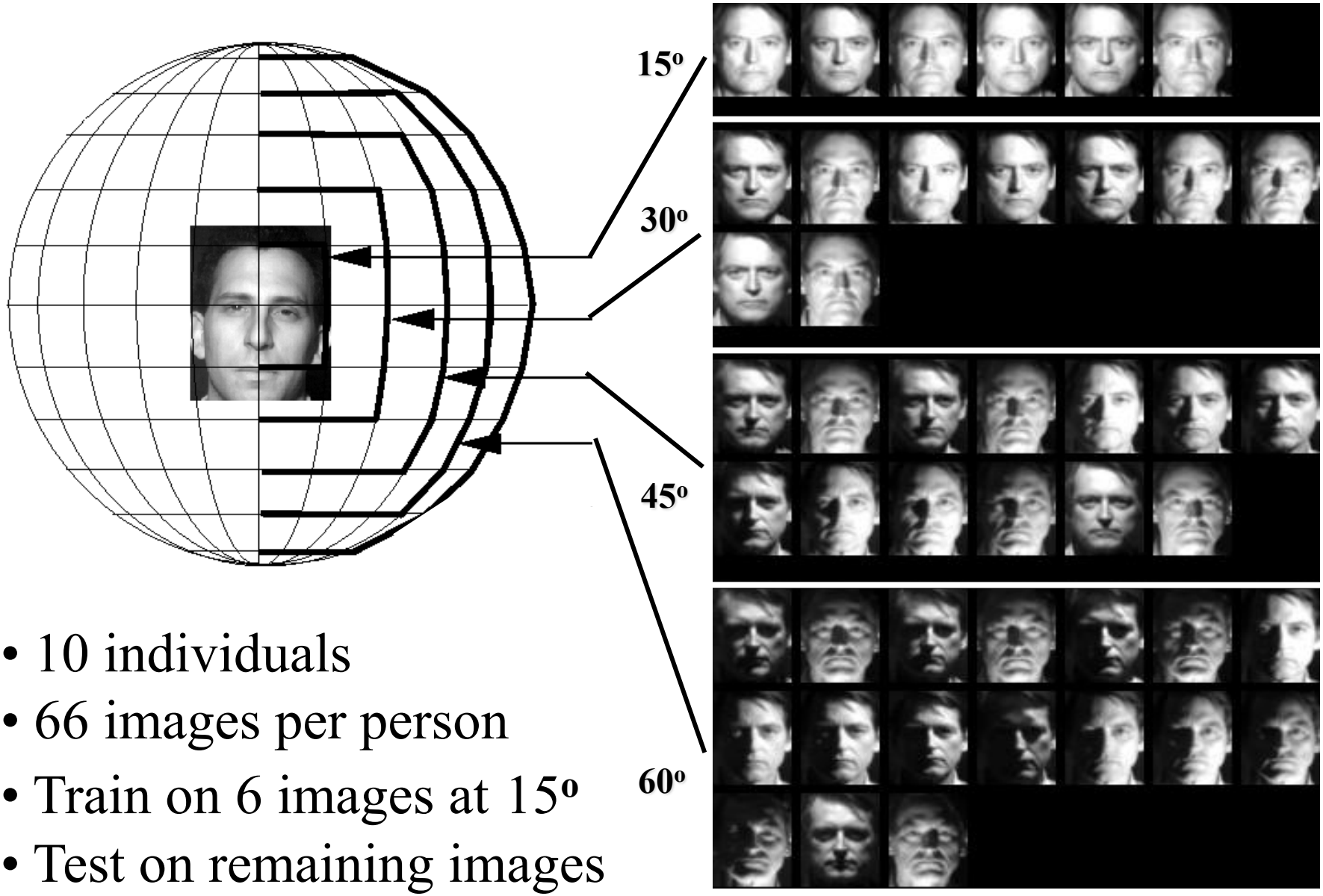
$$W_{PCA} = \arg\max_{W} \left| W^T S_T W \right|$$

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

- Since S_W is rank N-c, project training set to subspace spanned by first *N-c* principal components of the training set.
- Apply FLD to *N-c* dimensional subspace yielding *c-1* dimensional feature space.

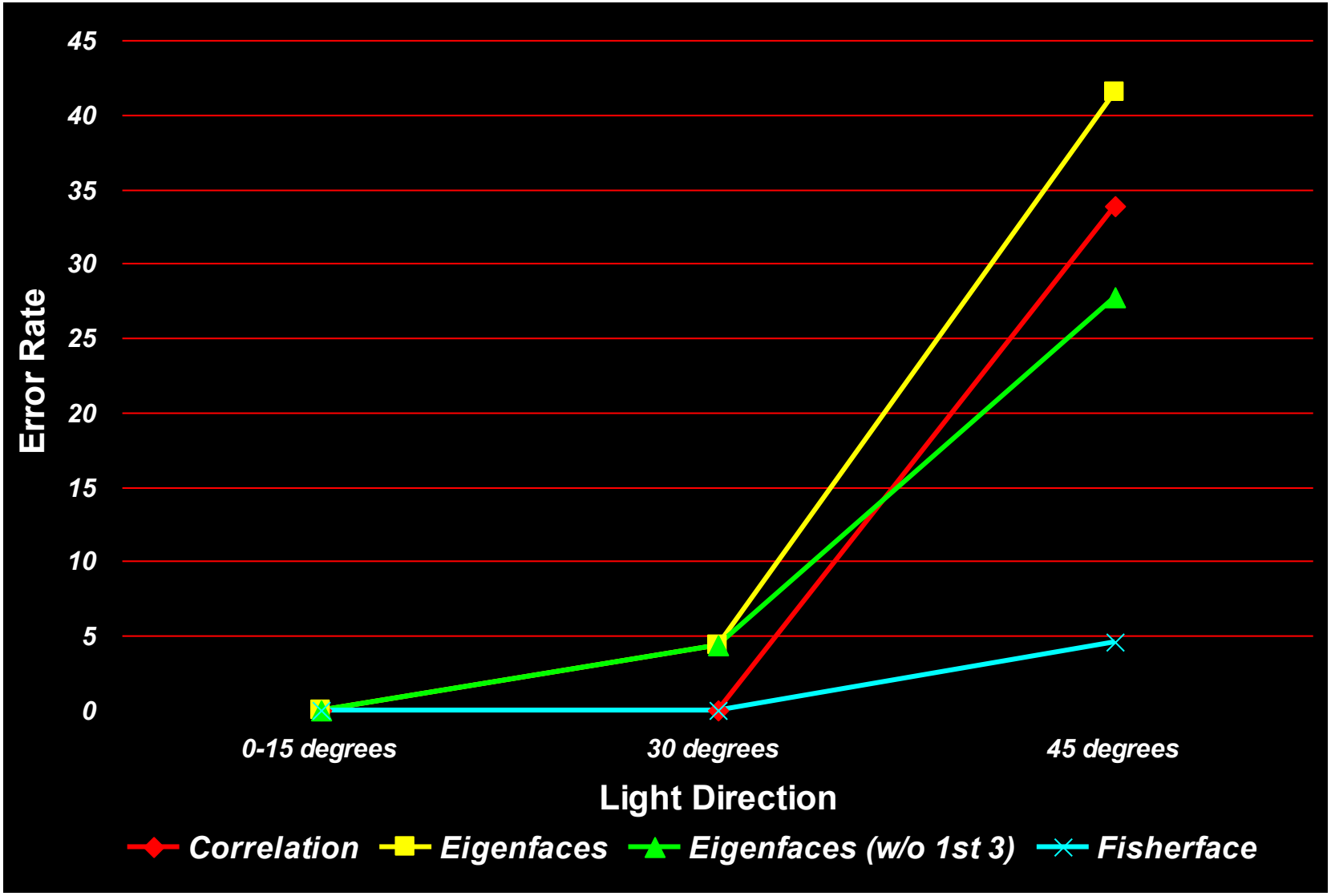
- Fisher's Linear Discriminant projects away the within-class variation (lighting, expressions) found in training set.
- Fisher's Linear Discriminant preserves the separability of the classes.

Harvard Face Database

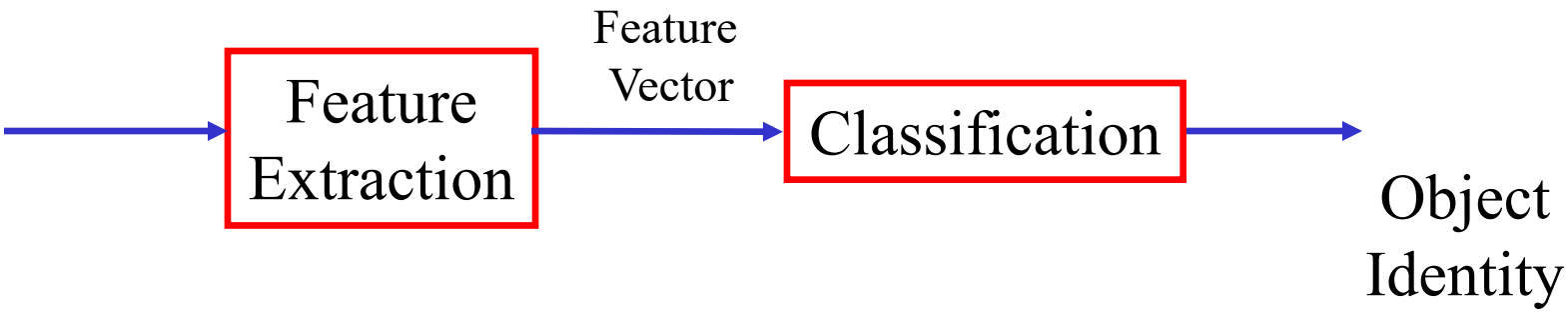


- 10 individuals
- 66 images per person
- Train on 6 images at 15°
- Test on remaining images

Recognition Results: Lighting Extrapolation



Where have we been so far



- Supervised classification
  - Feature space
  - Nearest Neighbor (kth nearest neighbor)
  - Bayesian (MAP) classifier
- Curse of dimensionality
- Dimensionality reduction
  - Principal component analysis
  - Fisher's linear discriminant

Next Lecture

- Neural networks