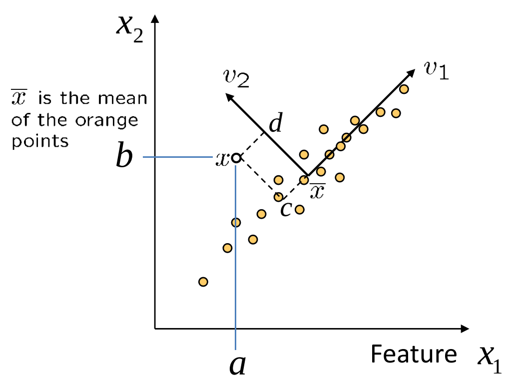
Principle Component Analysis

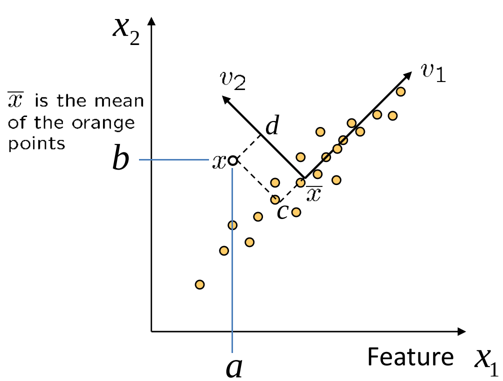
Dimensionality Reduction

$\mathbf{x}:(a,b)\rightarrow(c,d)$



- We can represent the orange points with *only* their $\mathbf{v_1}$ coordinates
  - since $\mathbf{v_2}$ coordinates are all essentially 0
- This makes it much cheaper to store and compare points
- A bigger deal for higher dimensional problems

Projection



convert **x** into **v₁, v₂** coordinates

$$\mathbf{x} \rightarrow ((\mathbf{x}-\overline{x})\cdot\mathbf{v_1}, (\mathbf{x}-\overline{x})\cdot\mathbf{v_2})$$

For example; $\mathbf{x}=(1{,}5), \overline{x}=(2{,}2)$

$v_1=[1/\sqrt{2}\ \ 1/\sqrt{2}]^T$

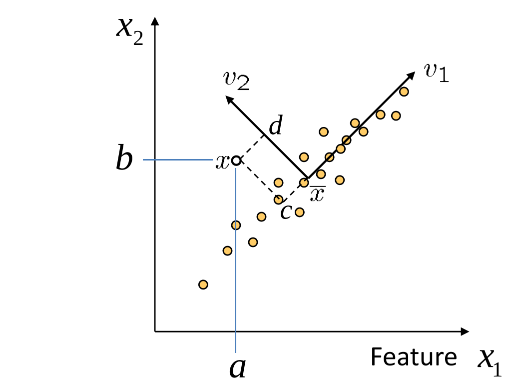
$v_2=[-1/\sqrt{2}\ \ 1/\sqrt{2}]^T$

$$\mathbf{x}\rightarrow\left(\begin{bmatrix}-1\\3\end{bmatrix}\cdot\begin{bmatrix}1/\sqrt{2}\\1/\sqrt{2}\end{bmatrix},\begin{bmatrix}-1\\3\end{bmatrix}\cdot\begin{bmatrix}-1/\sqrt{2}\\1/\sqrt{2}\end{bmatrix}\right)$$

$$\rightarrow\left(\sqrt{2},2\sqrt{2}\right)$$

PCA

Principle Component Analysis



Consider the variation along direction **v** among all of the orange points:

$$var(\mathbf{v}) = \sum_{\text{orange point } \mathbf{x}} \|(\mathbf{x}-\overline{\mathbf{x}})^{\mathbf{T}}\cdot\mathbf{v}\|^2$$

What unit vector **v** minimizes *var*?

$$\mathbf{v_2} = min_{\mathbf{v}}\ \{var(\mathbf{v})\}$$

What unit vector **v** maximizes *var*?

$$\mathbf{v_1} = max_{\mathbf{v}}\ \{var(\mathbf{v})\}$$

(under the constraint $\|\mathbf{v}\|=1$)

PCA
Principle Component Analysis

$$
\begin{aligned}
var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x}-\bar{\mathbf{x}})^{\mathrm{T}} \cdot \mathbf{v}\| \\
&= \sum_{\mathbf{x}} \mathbf{v}^{\mathrm{T}}(\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^{\mathrm{T}}\mathbf{v} \\
&= \mathbf{v}^{\mathrm{T}}\left[\sum_{\mathbf{x}}(\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^{\mathrm{T}}\right]\mathbf{v} \\
&= \mathbf{v}^{\mathrm{T}}\mathbf{A}\mathbf{v} \quad \text{where } \boxed{\mathbf{A} = \sum_{\mathbf{x}}(\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^{\mathrm{T}}}
\end{aligned}
$$

Covariance Matrix

Solution: $\mathbf{v_1}$ is eigenvector of **A** with *largest* eigenvalue
$\mathbf{v_2}$ is eigenvector of **A** with *smallest* eigenvalue

Eigenvectors & Eigenvalues

Eigenvalue problem : Find vector $\mathbf{v}$ and λ that make

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v} \quad \text{subject to} \quad \mathbf{v} \neq 0$$

$\mathbf{v}$ is call eigenvector of matrix $\mathbf{A}$

λ is call eigenvalue of matrix $\mathbf{A}$

- Can be solve by find λ that make $\det(\mathbf{A}-\lambda I)=0$ then find $\mathbf{v}$ that make $(\mathbf{A}-\lambda I)\mathbf{v}=\mathbf{0}$
- $\mathbf{A} : N \times N \Rightarrow$ We cam obtain N possible solutions for eigenvector-eigenvalue pairs (v_i, λ_i)

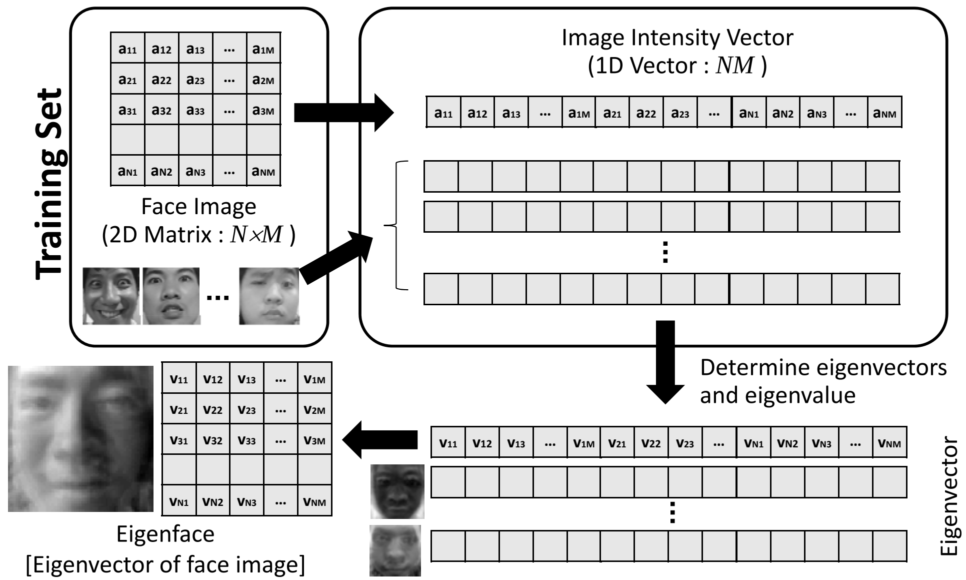
PCA
Principle Component Analysis

- Suppose each data point is N-dimensional
  - Same procedure applies:

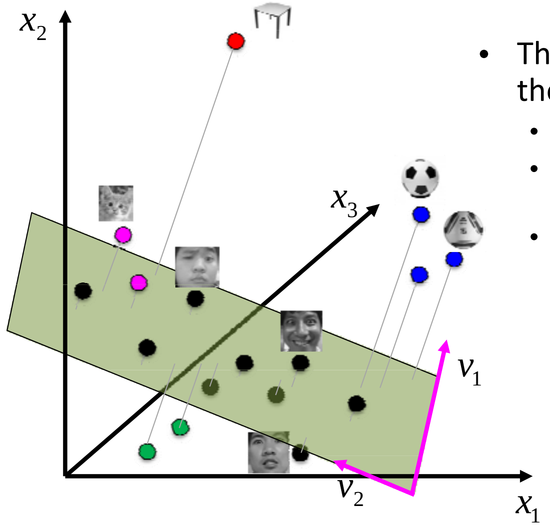
$$
\begin{aligned}
var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x}-\bar{\mathbf{x}})^{\mathrm{T}} \cdot \mathbf{v}\| \\
&= \mathbf{v}^{\mathrm{T}}\mathbf{A}\mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}}(\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^{\mathrm{T}}
\end{aligned}
$$

  - The eigenvectors of **A** define a new coordinate system
    - eigenvector with largest eigenvalue captures the most variation among training vectors **x**
    - eigenvector with smallest eigenvalue has least variation
  - We can compress the data by only using the top few eigenvectors
    - corresponds to choosing a "linear subspace"
      - represent points on a line, plane, or "hyper-plane"
    - these eigenvectors are known as the ***principal components***

Eigenfaces



Eigenfaces

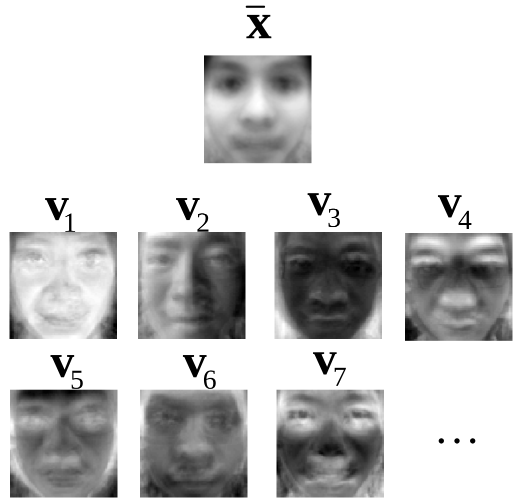


- The set of faces is a "subspace" of the set of images
  - Suppose it is K dimensional
  - We can find the best subspace using PCA
  - This is like fitting a "hyper-plane" to the set of faces
    - spanned by vectors $\mathbf{v_1}, \mathbf{v_2}, ..., \mathbf{v_K}$
    - any face

$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots + a_k\mathbf{v}_k$$

Eigenface Extraction

- PCA extracts the eigenvectors of **A**
  - Gives a set of vectors $\mathbf{v_1}$, $\mathbf{v_2}$, $\mathbf{v_3}$, ...
  - Each one of these vectors is a direction in face space
    - what do these look like?

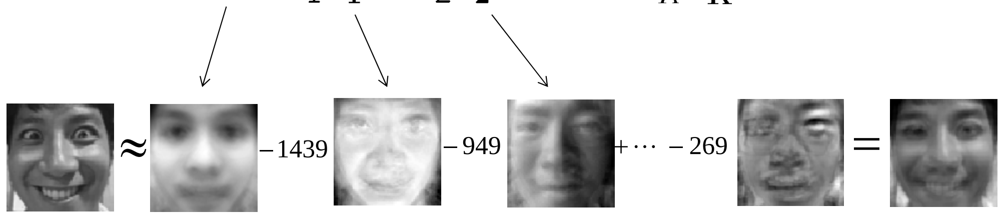


Eigenface Features
Projecting image onto the Eigenfaces

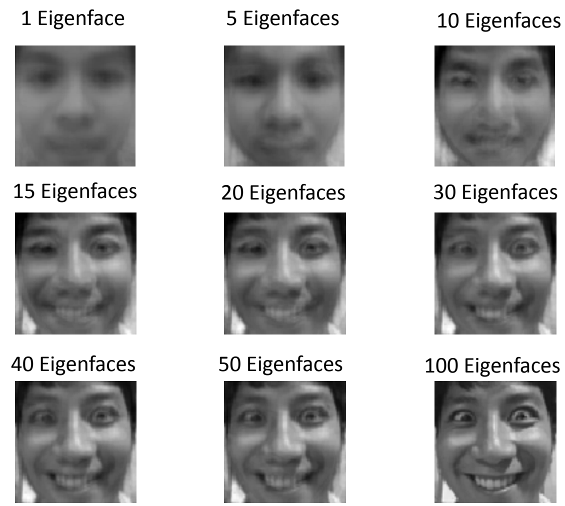
- The eigenfaces $\mathbf{v_1}, ..., \mathbf{v_K}$ span the space of faces
  - A face is converted to eigenface coordinates by

$$\mathbf{x} \rightarrow (\underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v}_1}_{a_1},\ \underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v}_2}_{a_2}, \ldots,\ \underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v}_K}_{a_K})$$

$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots + a_K\mathbf{v}_K$$

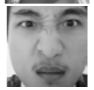


Eigenface Features
Projecting image onto the Eigenfaces



Eigenface Features

Projecting image onto the Eigenfaces

	a1	a2	a3	a4	a5	a6
	-1439	-949	711	21	-736	-1051
	-1376	-937	819	-214	-707	-1024
	1383	1092	149	-10	280	686
	798	823	569	1140	-192	676

Face Detection and Recognition using Eigenfaces

- Algorithm
  1. Process the image database (set of images with labels)
     - Run PCA—compute eigenfaces
     - Calculate the K coefficients for each image
  2. Given a new image (to be recognized) **x**, calculate K coefficients

$$\mathbf{x} \rightarrow (a_1, a_2, \ldots, a_K)$$

  3. Detect if x is a face

$$\|\mathbf{x} - (\overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K})\| < \text{threshold}$$

  4. If it is a face, who is it?
     - Find closest labeled face in database
       - nearest-neighbor in K-dimensional space