Project

- project groups
  - groups of 3–4
  - if needed, feel free to use "**Search for Teammates!**" feature on Piazza (pinned)
  - send me an email (mvasconcelos@eng.ucsd.edu) stating who are the group **members** (please use your official UCSD name) as soon as you know it, with deadline Tuesday, 1/18

- project proposal
  - due Tuesday, 2/1 @ 11:59pm
  - one–page maximum stating:
    - problem
    - data you will use
    - draft of proposed solution (can be updated later)
    - experiments you will run (can be updated later)
    - references (you can use an additional page for this)

ECE 271B – Winter 2022

PCA and LDA

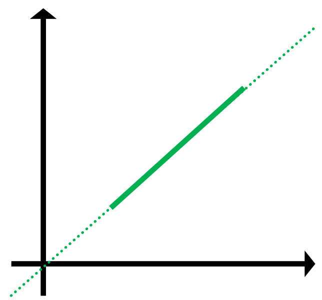
Disclaimer:
This class will be recorded
and made available to students asynchronously.

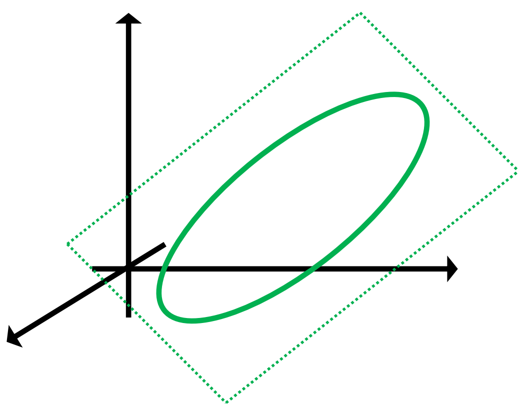
Manuela Vasconcelos

ECE Department, UCSD

Principal Component Analysis

- basic idea:
  - if the data lives in a subspace, it is going to look **very flat** when viewed from the full space, e.g.

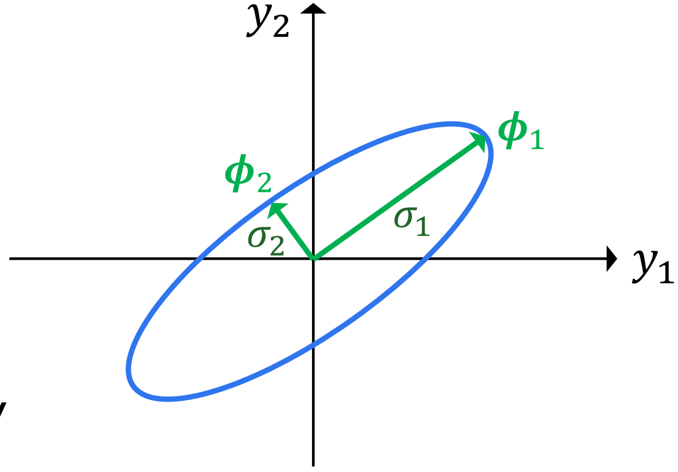




  - this means that if we fit a **Gaussian** to the data, the equiprobability contours are going to be **highly skewed** **ellipsoids**

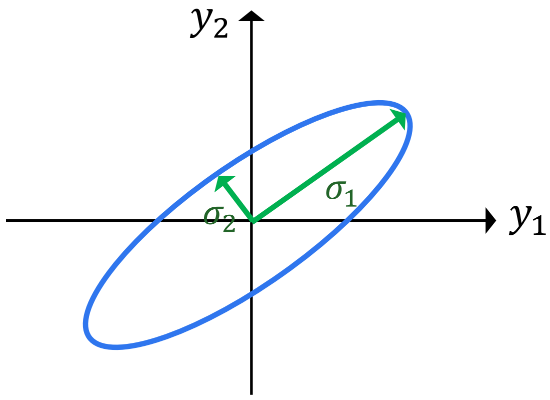
Principal Component Analysis (Learning)

- if $\mathbf{y}$ is Gaussian with covariance $\boldsymbol{\Sigma}$, the equiprobability contours are the ellipses whose
  - principal components $\boldsymbol{\phi}_i$ are the eigenvectors of $\boldsymbol{\Sigma}$
  - principal lengths σ_i are the eigenvalues of $\boldsymbol{\Sigma}$

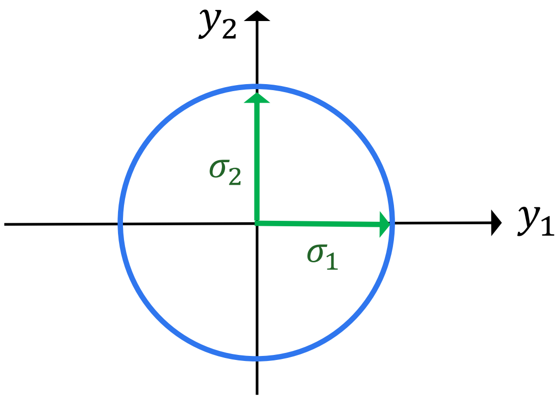


- by computing the eigenvalues, we know if the data is flat

$\sigma_1 \gg \sigma_2$: flat



$\sigma_1 = \sigma_2$: not flat



Principal Component Analysis (Learning)

Given sample $\mathcal{D} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$, $\mathbf{x}_i \in \mathbb{R}^d$

- compute sample mean: $\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_i \mathbf{x}_i$
- compute sample covariance: $\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_i (\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^T$
- compute eigenvalues and eigenvectors of $\hat{\boldsymbol{\Sigma}}$

$$\hat{\boldsymbol{\Sigma}} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^T \qquad \boldsymbol{\Lambda} = diag\left(\sigma_1^2, \cdots, \sigma_d^2\right) \qquad \boldsymbol{\Phi}\boldsymbol{\Phi}^T = \mathbf{I}$$

- order eigenvalues: $\sigma_1^2 > \cdots > \sigma_n^2$
- if, for a certain k, $\sigma_k \ll \sigma_1$, eliminate the eigenvalues and eigenvectors above k

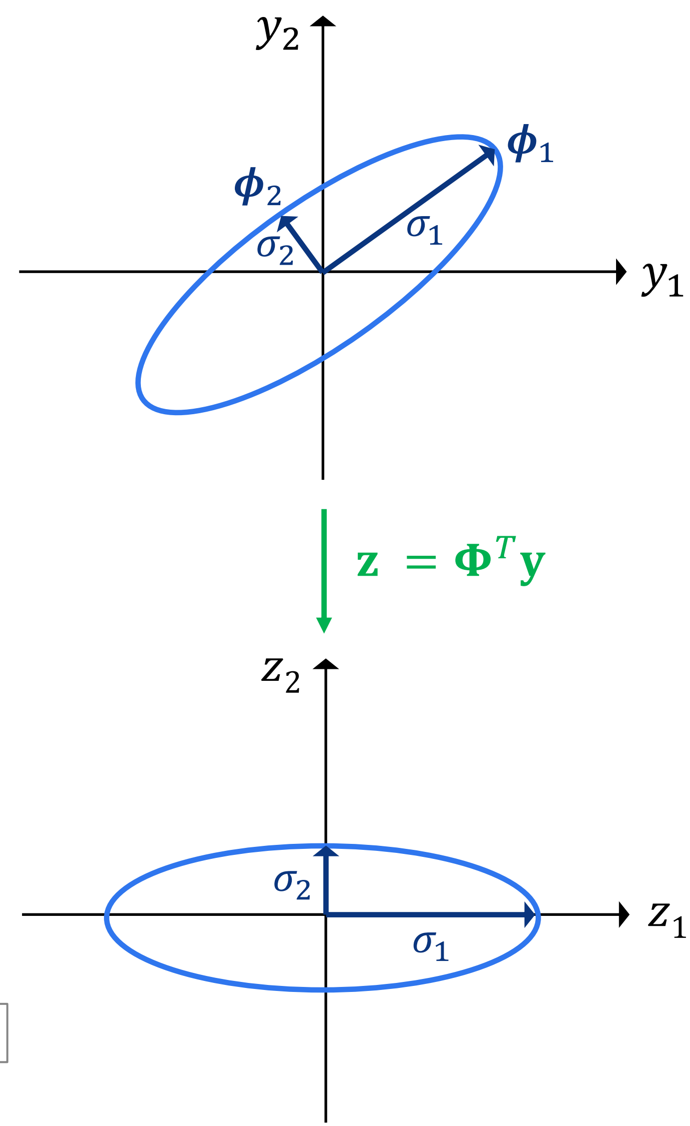
Principal Component Analysis

- Given a vector $\mathbf{y}$ in the original space $\mathbb{R}^d$
  - we can obtain the vector $\mathbf{z}$ by applying the transformation $\mathbf{\Phi}^{-1}$
  - since $\mathbf{\Phi}$ is orthogonal, this is just the transpose: $\mathbf{\Phi}^{-1} = \mathbf{\Phi}^T$

    $$\mathbf{\Phi} = \begin{bmatrix} | & & | \\ \boldsymbol{\phi}_1 & \cdots & \boldsymbol{\phi}_d \\ | & & | \end{bmatrix}$$

    note: $\mathbf{\Phi}^T$ is the matrix with principal components $\boldsymbol{\phi}_i$ as rows
  - elimination of components of small variance (eigenvalue) corresponds to eliminating all but k of these rows
  - in summary, PCA can be implemented with

$$\underbrace{\mathbf{z}}_{k-\text{dimensional}} = \underbrace{\begin{bmatrix} - & \boldsymbol{\phi}_1^T & - \\ & \vdots & \\ - & \boldsymbol{\phi}_k^T & - \end{bmatrix}}_{k \times d} \underbrace{\mathbf{y}}_{d-\text{dimensional}}$$



Principal Component Analysis

Given principal components $\boldsymbol{\phi}_i, i \in \{1, \cdots, k\}$, and a test sample $\mathcal{T} = \{\mathbf{t}_1, \cdots, \mathbf{t}_n\}$, $\mathbf{t}_i \in \mathbb{R}^d$

- subtract mean to each point: $\mathbf{t}'_i = \mathbf{t}_i - \widehat{\boldsymbol{\mu}}$
- project onto eigenvector space

$$\mathbf{y}_i = \mathbf{A}\,\mathbf{t}'_i \qquad \text{where} \qquad \mathbf{A} = \begin{bmatrix} - & \boldsymbol{\phi}_1^T & - \\ & \vdots & \\ - & \boldsymbol{\phi}_k^T & - \end{bmatrix}$$

- use $\mathcal{T}' = \{\mathbf{y}_1, \cdots, \mathbf{y}_n\}$ to do <u>all subsequent machine learning</u>

$$\underbrace{\mathbf{y}_i}_{k-\text{dimensional}} = \underbrace{\begin{bmatrix} - & \boldsymbol{\phi}_1^T & - \\ & \vdots & \\ - & \boldsymbol{\phi}_k^T & - \end{bmatrix}}_{k \times d} \underbrace{[\,\mathbf{t}_i - \widehat{\boldsymbol{\mu}}\,]}_{d-\text{dimensional}}$$

Principal Components

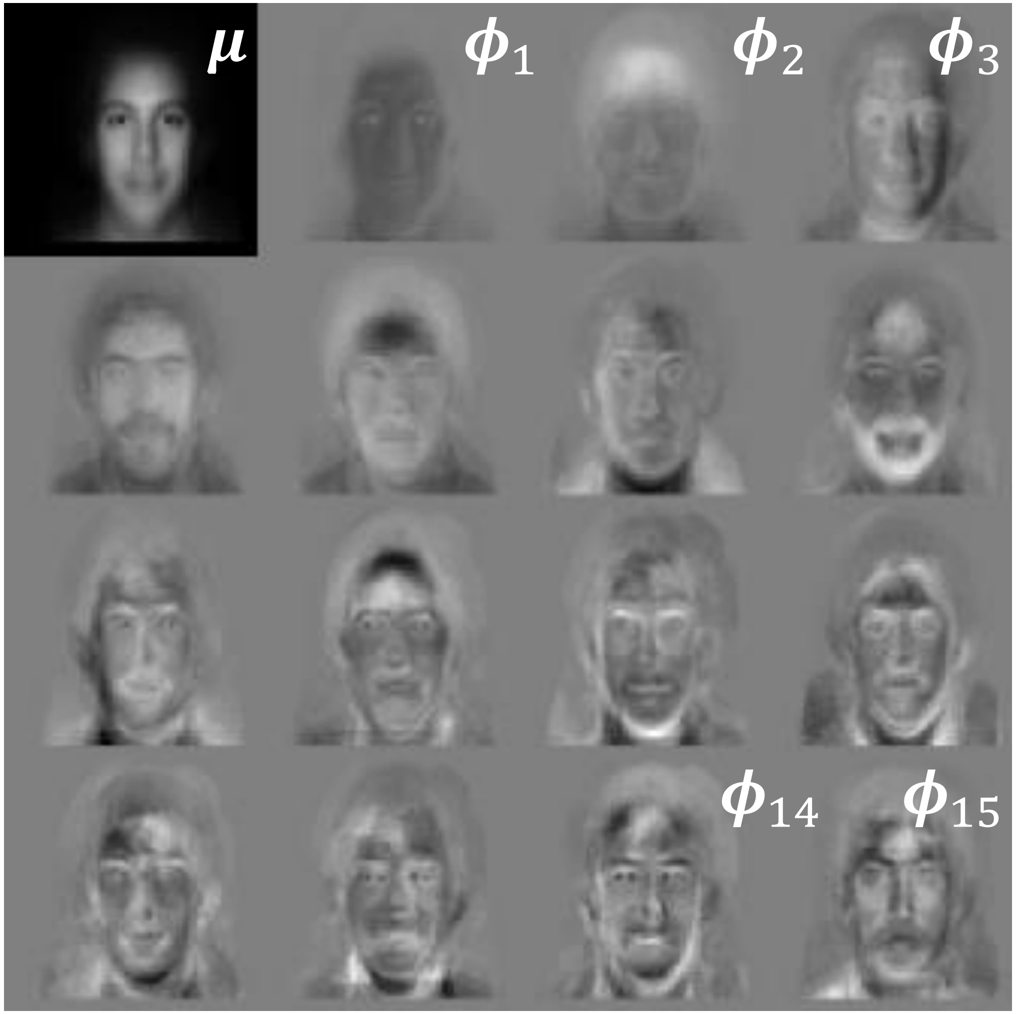
- what are they? in some cases it is possible to see
- example: eigenfaces
  - face recognition problem: can you identify who is the person in this picture?
  - training:
    - assemble examples from people's faces
    - compute the PCA basis
    - project each image into PCA space
    - use image projections to learn a classifier in the PCA space
  - recognition:
    - project image to classify into PCA space
    - apply the classifier in the PCA space

Principal Components

- face examples

Principal Components

- principal components (eigenfaces)
  - high–variance ones tend to have low–frequency
  - capture average face, illumination, etc.
  - at the intermediate–level, we have face detail
  - low–variance tends to be high–frequency noise



PCA by SVD

- there is an alternative manner to compute the principal components, based on Singular Value Decomposition (SVD)

- SVD:
  - any real $n \times m$ ($n > m$) matrix $\mathbf{A}$ can be decomposed as

$$\mathbf{A} = \mathbf{M}\,\boldsymbol{\Pi}\,\mathbf{N}^T$$

  - $\mathbf{M}$ is a $n \times m$ **column orthonormal** matrix of left singular vectors (columns of $\mathbf{M}$)
  - $\boldsymbol{\Pi}$ is a $m \times m$ **diagonal** matrix of singular values
  - $\mathbf{N}^T$ is a $m \times m$ **row orthonormal** matrix of right singular vectors (columns of $\mathbf{N}$)

$$\mathbf{M}^T\mathbf{M} = \mathbf{I} \qquad \mathbf{N}^T\mathbf{N} = \mathbf{I}$$

PCA by SVD

- to relate this to PCA,
  we consider the data−matrix

$$\mathbf{X} = \begin{bmatrix} | & & | \\ \mathbf{x}_1 & \cdots & \mathbf{x}_n \\ | & & | \end{bmatrix}, \ \mathbf{x}_i \in \mathbb{R}^d$$

- the sample mean is

$$\boldsymbol{\mu} = \frac{1}{n}\sum_i \mathbf{x}_i = \frac{1}{n}\begin{bmatrix} | & & | \\ \mathbf{x}_1 & \cdots & \mathbf{x}_n \\ | & & | \end{bmatrix}\underbrace{\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}}_{d \times 1} = \frac{1}{n}\mathbf{X}\mathbf{1}$$

and we can center the data by subtracting the mean to each column of $\mathbf{X}$ – this is the centered data−matrix

$$\boxed{\mathbf{X}_c} = \begin{bmatrix} | & & | \\ \mathbf{x}_1 & \cdots & \mathbf{x}_n \\ | & & | \end{bmatrix} - \begin{bmatrix} | & & | \\ \boldsymbol{\mu} & \cdots & \boldsymbol{\mu} \\ | & & | \end{bmatrix}$$

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

PCA by SVD

- the sample covariance is

$$\boxed{\boldsymbol{\Sigma} = \frac{1}{n}\sum_i (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T = \frac{1}{n}\sum_i \mathbf{x}_i^c(\mathbf{x}_i^c)^T}$$

where $\mathbf{x}_i^c$ is the i^{th} column of $\mathbf{X}_c$

- this can be written as

$$\boxed{\boldsymbol{\Sigma} = \frac{1}{n}\begin{bmatrix} | & & | \\ \mathbf{x}_1^c & \cdots & \mathbf{x}_n^c \\ | & & | \end{bmatrix}\begin{bmatrix} - & \mathbf{x}_1^c & - \\ & \vdots & \\ - & \mathbf{x}_n^c & - \end{bmatrix} = \frac{1}{n}\mathbf{X}_c\mathbf{X}_c^T}$$

and the centered data–matrix contains <u>all</u> the covariance information

PCA by SVD

- the centered data−matrix

$$\mathbf{X}_c^T = \begin{bmatrix} - & \mathbf{x}_1^c & - \\ & \vdots & \\ - & \mathbf{x}_n^c & - \end{bmatrix}$$

is real $n \times d$. Assuming $n > d$, it has SVD decomposition

$$\mathbf{X}_c^T = \mathbf{M}\,\mathbf{\Pi}\,\mathbf{N}^T$$

$\mathbf{M}$ – orthonormal matrix
$\mathbf{\Pi}$ – diagonal matrix
$\mathbf{N}^T$ – orthonormal matrix

$$\mathbf{M}^T\mathbf{M} = \mathbf{I} \qquad \mathbf{N}^T\mathbf{N} = \mathbf{I}$$

and

$$\mathbf{\Sigma} = \frac{1}{n}\mathbf{X}_c\mathbf{X}_c^T = \frac{1}{n}\mathbf{N}\,\mathbf{\Pi}\,\mathbf{M}^T\mathbf{M}\,\mathbf{\Pi}\,\mathbf{N}^T = \frac{1}{n}\mathbf{N}\,\mathbf{\Pi}^2\mathbf{N}^T$$

$$\boxed{\mathbf{\Sigma} = \mathbf{N}\left(\frac{1}{n}\mathbf{\Pi}^2\right)\mathbf{N}^T}$$

orthonormal

diagonal

orthonormal

PCA by SVD

$$\boxed{\mathbf{\Sigma} = \mathbf{N}\left(\frac{1}{n}\mathbf{\Pi}^2\right)\mathbf{N}^T}$$

- noting that $\mathbf{N}$ is $d \times d$ and orthonormal and $\mathbf{\Pi}^2$ is diagonal, this is just the eigenvalue decomposition of $\mathbf{\Sigma}$

- it follows that
  - the eigenvectors of $\mathbf{\Sigma}$ are the **columns** of $\mathbf{N}$
  - the eigenvalues of $\mathbf{\Sigma}$ are

$$\boxed{\sigma_i = \frac{1}{n}\pi_i^2}$$

- this gives an **<u>alternative</u> algorithm** for PCA

PCA by SVD

$$\mathbf{X} = \begin{bmatrix} | & & | \\ \mathbf{x}_1 & \cdots & \mathbf{x}_n \\ | & & | \end{bmatrix}$$

- computation of PCA by SVD

Given $\mathbf{X}$ with one example per column

1) create the centered data−matrix

$$\boxed{\mathbf{X}_c^T = \left(\mathbf{I} - \frac{1}{n}\,\mathbf{1}\mathbf{1}^T\right)\mathbf{X}^T}$$

2) compute its SVD

$$\boxed{\mathbf{X}_c^T = \mathbf{M}\,\mathbf{\Pi}\,\mathbf{N}^T}$$

3) principal components are columns of $\mathbf{N}$, eigenvalues are

$$\boxed{\sigma_i = \frac{1}{n}\pi_i^2}$$

Limitations of PCA

- PCA is not optimal for classification
  - note that there is no mention of the **class label** in the definition of PCA
  - keeping the dimensions of largest energy (variance) is a good idea, but not always enough
  - certainly improves the density estimation since space has **smaller dimension**
  - but could be unwise from a classification point of view
  - the **discriminant dimensions** could be **thrown out**
- it is not hard to construct examples where PCA is the worst possible thing we could do

Limitations of PCA: Example

- consider a problem with
  - two $n-$ D Gaussian classes with covariance $\mathbf{\Sigma} = \sigma^2\mathbf{I}$, $\sigma^2 = 10$

$$\mathbf{X} \sim N(\boldsymbol{\mu}_i, 10\mathbf{I})$$

  - we add an extra variable which is the class Y label itself

$$\mathbf{X}' = [\boldsymbol{X}, Y]$$

  - each of the n dimensions of $\mathbf{X}$ has variance 10
  - what about the new dimension Y?
    - assuming that $P_Y(0) = P_Y(1) = 0.5$

$$E[Y] = 0.5 \times 0 + 0.5 \times 1 = 0.5$$

$$\text{var}[Y] = 0.5 \times (0 - 0.5)^2 + 0.5 \times (1 - 0.5)^2 = 0.25 < 10$$

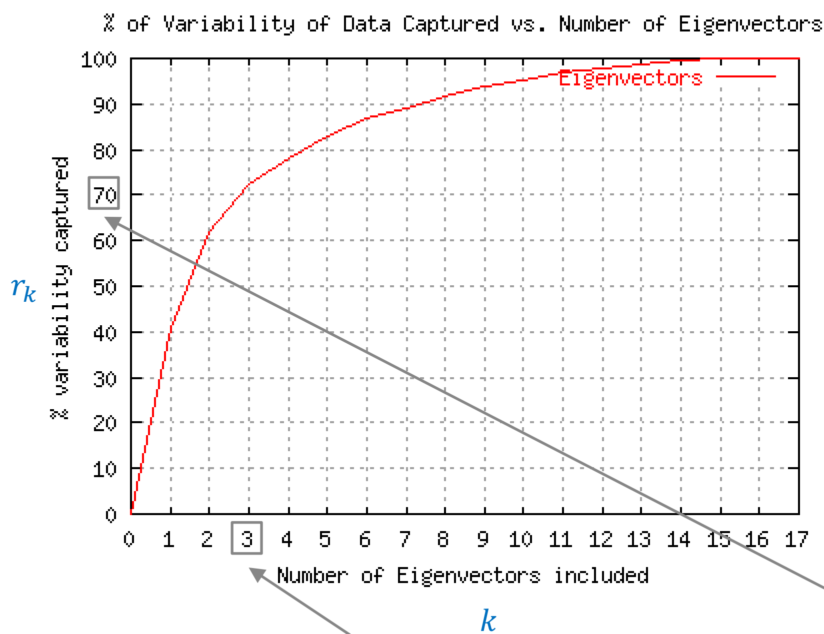
  - the new dimension Y has the <u>smallest</u> variance and is the <u>first</u> to be discarded by PCA!

Limitations of PCA

- this is
  - a very contrived example
  - but shows that PCA can **throw away** all the discriminant info
- does this mean you should **never** use PCA?
  - **no**, typically it is a **good** method to find a suitable subset of variables as long as you are **not** too **greedy**
    - e.g. if you start with $n = 100$ and know that there are only 5 variables of interest
    - picking the top 20 PCA components is likely to keep the desired 5
    - your classifier will be much better than for $n = 100$, probably not much worse than the one with the best 5 features
- is there a **rule of thumb** for finding the **number of PCA components**?

Principal Component Analysis

- a natural measure is to pick the eigenvectors that **explain $p\%$ of the data variability**
  - can be done by plotting the ratio r_k as a function of k



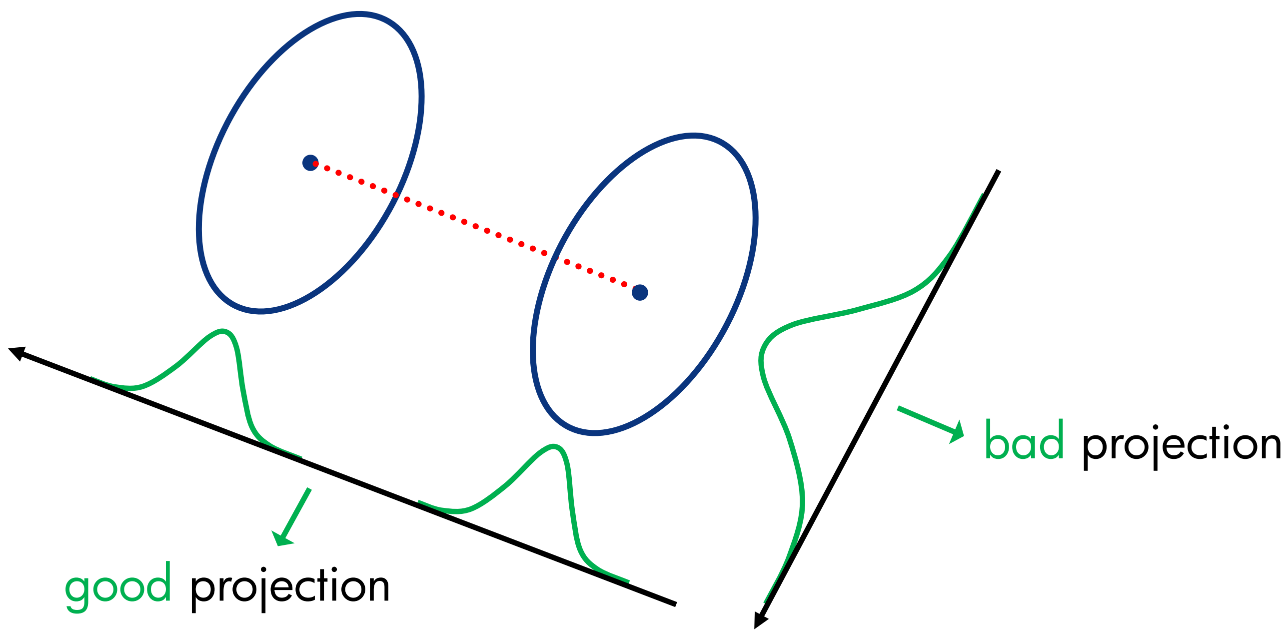
$$r_k = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{n} \sigma_i^2}$$

  - e.g. we need 3 eigenvectors to cover 70% of the variability of this dataset

Linear Discriminant Analysis

- what if we really need to find the <u>best</u> **features**?
  - **harder question**, usually <u>impossible</u> with <u>simple</u> methods
  - however, there are **better** methods at **finding discriminant directions**

- one good example is Linear Discriminant Analysis (LDA)
  - the idea is to **find the line that best separates the two classes**



Linear Discriminant Analysis

- we have two classes $Y \in \{0,1\}$ such that

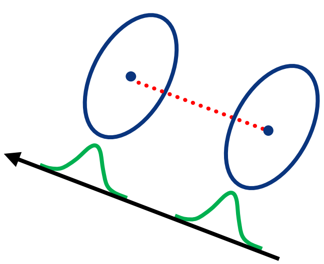
$$E_{\mathbf{X}|Y}[\mathbf{x}|y=i] = \boldsymbol{\mu}_i \qquad E_{\mathbf{X}|Y}[(\mathbf{x}-\boldsymbol{\mu}_i)(\mathbf{x}-\boldsymbol{\mu}_i)^T|y=i] = \boldsymbol{\Sigma}_i$$

and we want to find the line, i.e. the direction $\mathbf{w}$, such that the projection

$$\boxed{z = \mathbf{w}^T\mathbf{x}}$$

best separates the classes

- one possibility would be to maximize the distance between the means after the projection



$$\left(E_{Z|Y}[z|y=1] - E_{Z|Y}[z|y=0]\right)^2$$
$$= \left(E_{\mathbf{X}|Y}[\mathbf{w}^T\mathbf{x}\ |y=1] - E_{\mathbf{X}|Y}[\mathbf{w}^T\mathbf{x}\ |y=0]\right)^2$$
$$= (\mathbf{w}^T[\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0])^2$$

Linear Discriminant Analysis

$z = \mathbf{w}^T\mathbf{x}$

$\left(E_{Z|Y}[z|y=1] - E_{Z|Y}[z|y=0]\right)^2 = (\mathbf{w}^T[\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0])^2$

- however,

$$(\mathbf{w}^T[\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0])^2$$

  can be made arbitrarily large by simply scaling $\mathbf{w}$

- we are only interested in the **direction**, **not** the magnitude
- we need some type of **normalization**
- Fisher suggested

Fisher, R. A.; The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, v. 7, p. 179–188, 1936.

LDA
is also known as
Fisher's Linear Discriminant

measures separation between class means

measures variability inside the classes

$$\max \frac{\text{between class scatter}}{\text{within class scatter}} =$$

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

Linear Discriminant Analysis

$Y \in \{0,1\}$

$E_{\mathbf{X}|Y}[\mathbf{x}|y=i] = \boldsymbol{\mu}_i$

$E_{\mathbf{X}|Y}[(\mathbf{x}-\boldsymbol{\mu}_i)(\mathbf{x}-\boldsymbol{\mu}_i)^T|y=i] = \boldsymbol{\Sigma}_i$

$z = \mathbf{w}^T\mathbf{x}$

- we have already seen that

$$
\begin{aligned}
&\left(E_{Z|Y}[z|y=1] - E_{Z|Y}[z|y=0]\right)^2 \\
&\quad = (\mathbf{w}^T[\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0])^2 \\
&\quad = \mathbf{w}^T[\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0][\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0]^T\mathbf{w}
\end{aligned}
$$

- also

$$
\begin{aligned}
\mathrm{var}[z|y=i] &= E_{Z|Y}\left\{\left(z - E_{Z|Y}[z|y=i\,]\right)^2 \middle| y=i\right\} \\
&= E_{\mathbf{X}|Y}\left\{\left(\mathbf{w}^T\mathbf{x} - E_{\mathbf{X}|Y}[\mathbf{w}^T\mathbf{x}|y=i\,]\right)^2 \middle| y=i\right\} \\
&= E_{\mathbf{X}|Y}\{(\mathbf{w}^T[\mathbf{x}-\boldsymbol{\mu}_i])^2 |\, y=i\} \\
&= E_{\mathbf{X}|Y}\{\mathbf{w}^T[\mathbf{x}-\boldsymbol{\mu}_i][\mathbf{x}-\boldsymbol{\mu}_i]^T\mathbf{w} \,|\, y=i\} \\
&= \mathbf{w}^T E_{\mathbf{X}|Y}\{[\mathbf{x}-\boldsymbol{\mu}_i][\mathbf{x}-\boldsymbol{\mu}_i]^T \,|\, y=i\}\,\mathbf{w} \\
&= \mathbf{w}^T\boldsymbol{\Sigma}_i\mathbf{w}
\end{aligned}
$$

Linear Discriminant Analysis

- and

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

$$= \frac{\mathbf{w}^T(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\mathbf{w}}{\mathbf{w}^T(\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)\mathbf{w}}$$

$\mathrm{var}[z|y=i] = \mathbf{w}^T\boldsymbol{\Sigma}_i\mathbf{w}$

- which can be written as

$$\boxed{J(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}}}$$

between class scatter
measures separation between class means

$$\mathbf{S}_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T$$

$$\mathbf{S}_W = \boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1$$

within class scatter
measures variability inside the classes

Linear Discriminant Analysis

- **maximizing** the ratio

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

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

$$\boxed{\max_{\mathbf{w}} \ \mathbf{w}^T \mathbf{S}_B \mathbf{w}} \quad \text{subject to} \quad \boxed{\mathbf{w}^T \mathbf{S}_W \mathbf{w} = K}$$

    and can be accomplished using Lagrange optimization

  - define the Lagrangian

$$\boxed{L = \mathbf{w}^T \mathbf{S}_B \mathbf{w} - \lambda(\mathbf{w}^T \mathbf{S}_W \mathbf{w} - K)}$$

    and maximize with respect to $\mathbf{w}$

Linear Discriminant Analysis

- setting the gradient of

$$L = \mathbf{w}^T \mathbf{S}_B \mathbf{w} - \lambda(\mathbf{w}^T \mathbf{S}_W \mathbf{w} - K) = \mathbf{w}^T (\mathbf{S}_B - \lambda \mathbf{S}_W)\mathbf{w} + \lambda K$$

  with respect to $\mathbf{w}$ to zero, we get

$$\nabla_{\mathbf{w}} L = 2(\mathbf{S}_B - \lambda \mathbf{S}_W)\mathbf{w} = 0$$

  or

$$\boxed{\mathbf{S}_B \mathbf{w} = \lambda \mathbf{S}_W \mathbf{w}}$$

- this is a generalized eigenvalue problem

- the solution is easy when $\boxed{\mathbf{S}_W^{-1} = (\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1)^{-1} \text{ exists}}$

$\mathbf{S}_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T$
$\mathbf{S}_W = \boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1$

Linear Discriminant Analysis

$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$

- in this case, $\mathbf{w}$ is the eigenvector with largest eigenvalue of $\mathbf{S}_W^{-1}\mathbf{S}_B$

$\mathbf{S}_W^{-1} = (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)^{-1}$ exists

$\mathbf{S}_B\mathbf{w} = \lambda \mathbf{S}_W \mathbf{w}$

$$\mathbf{S}_W^{-1}\mathbf{S}_B\mathbf{w} = \lambda\mathbf{w}$$

- it can actually be computed by using the definition of $\mathbf{S}_B$

$$\mathbf{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\mathbf{w} = \lambda\mathbf{w}$$

- noting that $(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^T\mathbf{w} = \alpha$ is a scalar, this can be written as

$$\mathbf{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) = \frac{\lambda}{\alpha}\mathbf{w}$$

and, since we don't care about the magnitude of $\mathbf{w}$,

$$\boxed{\mathbf{w}^* = \mathbf{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) = (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)}$$

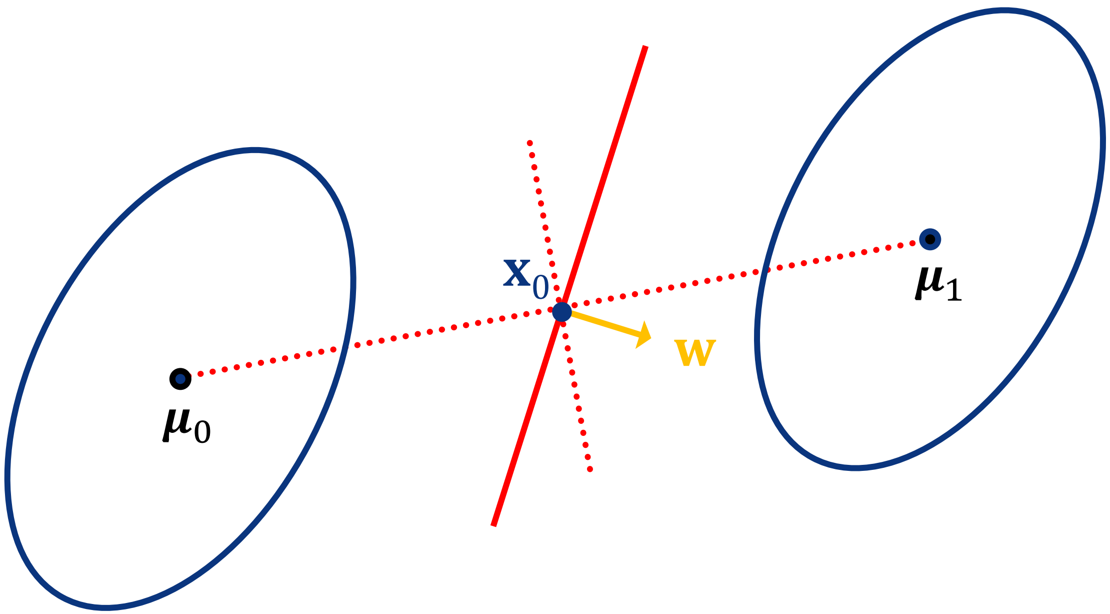
is the linear discriminant

$\mathbf{w}^* = \mathbf{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) = (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$

Linear Discriminant Analysis

- note that you have seen this before
  - for a classification problem with Gaussian classes of equal covariance $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_0$, the BDR boundary is the plane of normal

$$\mathbf{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$



Gaussian classes, equal covariance $\boldsymbol{\Sigma}$

  - if $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_0$, this is also the LDA solution

Linear Discriminant Analysis

- this gives two different **interpretations of LDA**

1. **classical Fisher interpretation**, which is just about separating data (assumes no probability model)

2. **Bayes decision rule**

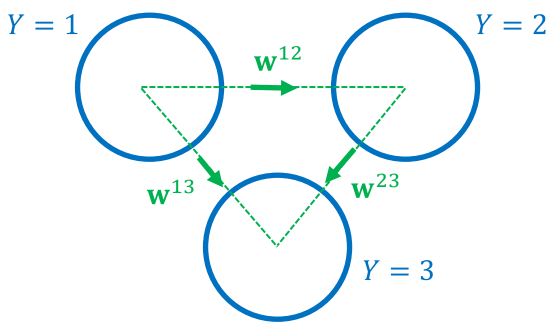
$$\boxed{i^* = \arg\max_i P(y = i|\mathbf{x})}$$

after approximating the data by two Gaussians with equal covariance

- hence, **LDA is usually better than PCA**
  - it explicitly optimizes discrimination
- but **not** necessarily good enough
  - if the Gaussian approximation is a poor one

Linear Discriminant Analysis

- what if there are more than two classes?
  - you simply compute the discriminants $\mathbf{w}^{ij}$ between all pairs of classes i and j
  - e.g. for $C = 3$ classes



$$\boxed{\mathbf{w}^{ij} = \left(\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j\right)^{-1}\left(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\right)}$$

  - note that there are $C(C-1)/2$ different pairs

- this **constrains** the dimension of LDA
  - you **cannot** simply pick what you want
  - the dimension after dimensionality reduction is $\boxed{C(C-1)/2}$

PCA + LDA

- the main difficulty of LDA is that computation of the **linear discriminant**

$$\mathbf{w}^* = (\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1)^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

  requires **matrix inversion**

- the inversion can be very error prone when the matrix $\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1$ is close to singular

- this happens when $\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1$ has eigenvalues close to zero

- to avoid this problem, it can be useful to adopt a **two–step** solution

  1. use PCA to eliminate the dimensions of small eigenvalues
  2. project into the remaining dimensions and apply LDA

- this is known as "PCA + LDA"