Lecture 10. PCA, SVD and Fisher Linear Discriminant

Prof. Alan Yuille

Spring 2014

Outline

1. Principal Component Analysis (PCA)
2. Singular Value Decomposition (SVD) – advanced material
3. Fisher Linear Discriminant

1 Principal Component Analysis (PCA)

One way to deal with the curse of dimensionality is to project data down onto a space of low dimensions, see figure (1). There are a number of different techniques for doing this. The most basic method is Principal Component Analysis (PCA) .

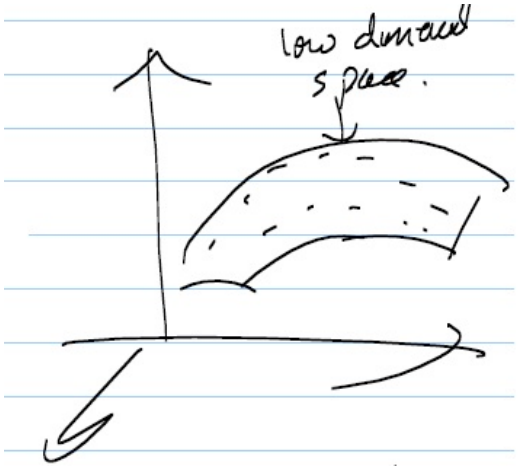

Figure 1:

We will use the following convention:

$\vec{\mu}^T\vec{\mu}$ is a scalar $\quad \mu_1^2 + \mu_2^2 + \cdots + \mu_D^2$

$\vec{\mu}\,\vec{\mu}^T$is a matrix $\begin{pmatrix} \mu_1^2 & \mu_1\mu_2 & \mu_1\mu_3 & \cdots \\ \vdots & \mu_2^2 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$

The data samples are $\vec{x}_1, \ldots, \vec{x}_N$ in a D-dimension space. First, compute their mean

$$\vec{\mu} = \frac{1}{N}\sum_{i=1}^{N}\vec{x}_i$$

and their covariance

$$\mathbf{K} = \frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T$$

Next, compute the eigenvalues and eigenvector of $\mathbf{K}$:
Solve $\mathbf{K}\vec{e} = \lambda\vec{e}$
$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$

Note: $\mathbf{K}$ is a symmetric matrix- so eigenvalues are real, eigenvectors are orthogonal. $\vec{e}_{nu} \cdot \vec{e}_\mu = 1$ if $\mu = \nu$, and $= 0$ otherwise. Also, by construction, the matrix $\mathbf{K}$ is positive semi-definite, so $\lambda_N \geq 0$ (i.e. no eigenvalues are negative).

PCA reduces the dimension by projection the data onto a space spanned by the eigenvectors $\vec{e}_i$ with $\lambda_i > T$, where T is a threshold. Let M eigenvectors be kept. Then, project data $\vec{x}$ onto the subspace spanned by the first M eigenvectors, after subtracting out the mean. Formally:

$$\vec{x} - \vec{\mu} = \sum_{\nu=1}^{D} a_\nu \vec{e}_\nu,$$

where the coefficients $\{a_\nu\}$ are given by

$$a_\nu = (\vec{x} - \vec{\mu}) \cdot \vec{e}_\nu$$

Note: orthogonality means $\vec{e}_\nu \cdot \vec{e}_\mu = \delta_{\nu\mu}$, which denotes the Kronecker delta.

Hence:

$$\vec{x} = \vec{\mu} + \sum_{\nu=1}^{D}\langle(\vec{x} - \vec{\mu}) \cdot \vec{e}_\nu\rangle\vec{e}_\nu$$

and there is no dimension reduction (no compression).

Then, approximate

$$\vec{x} \approx \vec{\mu} + \sum_{\nu=1}^{M} \langle (\vec{x} - \vec{\mu}) \cdot \vec{e}_\nu \rangle \vec{e}_\nu$$

This Projects the data into the M-dimension subspace of form:

$$\vec{\mu} + \sum_{\nu=1}^{M} b_\nu \vec{e}_\nu$$

See a 2-dimensions example in figure (2). The eigenvector of $\mathbf{K}$ corresponds to the second order movements of the data.

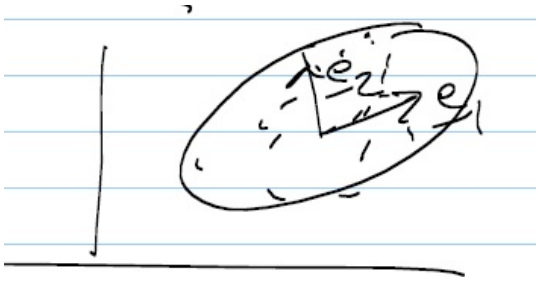

Figure 2: In two-dimensions, the eigenvectors give the principal axes of the data.

If the data lies (almost) on a straight line, then $\lambda_1 \gg 0, \lambda_2 \approx 0$, see figure (3).

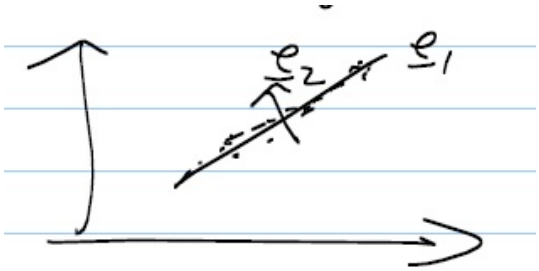

Figure 3: In two-dimensions, if the data lies along a line then $\lambda_1 > 0$ and $\lambda_2 \approx 0$.

1.1 PCA and Gaussian Distribution

PCA is equivalent to performing ML estimation of the parameters of a Gaussian distribution

$$p(\vec{x} \mid \vec{\mu}, \mathbf{\Sigma}) = \frac{1}{(2\pi)^{D/2}\sqrt{\det \mathbf{\Sigma}}} e^{-\frac{1}{z}(\vec{x}-\vec{\mu})^T \sum^{-} 1 (\vec{x}-\vec{\mu})}$$

to get $\hat{\vec{\mu}}, \hat{\mathbf{\Sigma}}$ by performing ML on $\prod_i p(\vec{x}_i \mid \vec{\mu}, \mathbf{\Sigma})$, and then throw away the directions where the standard deviation is small. ML gives $\vec{\mu} = \frac{1}{N}\sum_{i=1}^{N} \vec{x}_i$ and $\Sigma = \frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i - \vec{m}u)(\vec{x}_i - \vec{\mu})^T$. See Bishop's book for probabilistic PCA.

1.2 When is PCA appropriate?

PCA is almost always a good technique to try, because it is so simple. Obtain the eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$ and plot $f(M) = \sum_{i=1}^{M}\lambda_i / \sum_{i=1}^{N}\lambda_i$, to see how $f(M)$ increases with M and takes maximum value 1 at $M = D$. PCA is good if $f(M)$ asymptotes rapidly to 1. This happens if the first eigenvalues are big and the remainder are small. PCA is bad if all the eigenvalues are roughly equal. See examples of both cases in figure (4).

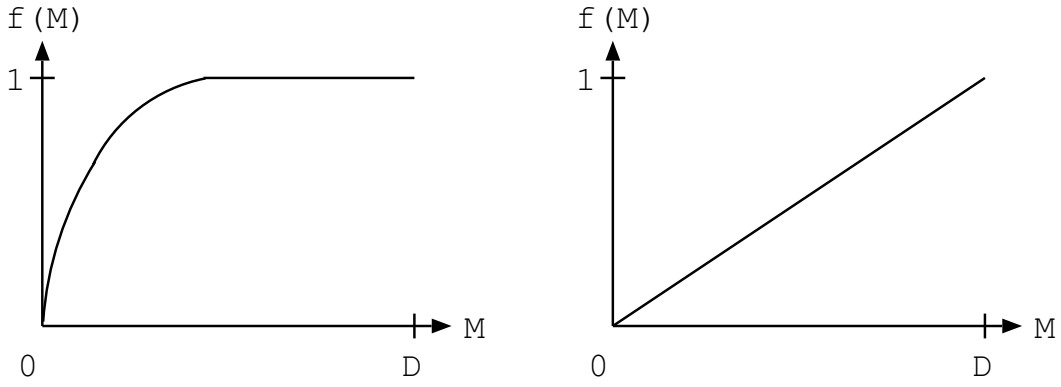

Figure 4: Left: eigenvalues asymptote rapidly to 1, this is good. Right: all eigenvalues are equally important, PCA is not appropriate here.

PCA would be bad in an example in which the data is a set of strings
$(1, 0, 0, 0, \dots) = \vec{x}_1$
$(0, 1, 0, 0, \dots) = \vec{x}_2$
$(0, 0, 0, 0, \dots, 0, 1) = \vec{x}_N$

Then, it can be computed that there is one zero eigenvalue of PCA. But all the other eigenvalues are not small. In general, PCA works best if there is a linear structure to the data. It works poorly if the data lies on a curved surface and not on a flat surface.

1.3 Interpretation of PCA

What is PCA doing? There are two equivalent ways to interpret PCA: (i) minimize the projection error, and (ii) maximize the variance of the projection.

Consider the variance of the data $\frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})^2$. It is independent of the projection. We can express $(\vec{x}_i - \vec{\mu})^2 = \sum_{i=1}^{D}\{(\vec{x}_i - \vec{\mu}) \cdot \vec{e}_\nu\}^2$, where the $\vec{e}_\nu$ are the eigenvectors of the

correlation. Hence,

$$\frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i-\vec{\mu})^2=\frac{1}{N}\sum_{i=1}^{N}\sum_{\nu=1}^{M}\{(\vec{x}_i-\vec{\mu})\cdot\vec{e}_\nu\}^2+\frac{1}{N}\sum_{i=1}^{N}\sum_{\nu=M+1}^{D}\{(\vec{x}_i-\vec{\mu})\cdot\vec{e}_\nu\}^2,$$

where the left-hand side is the variance of the data, the first term of the right-hand side is the variance of the data within the plane $\vec{e}_1,\cdots,\vec{e}_M$, and the last term is the projection error.

When $\vec{x}_i$ is projected to a point $\vec{x}_{i,p}=\vec{\mu}+\sum_{\nu=1}^{M}\{(\vec{x}_i-\vec{\mu})\cdot\vec{e}_\nu\}\vec{e}_\nu$, it has a projection error $\sum_{\nu=M+1}^{D}\{(\vec{x}_i-\vec{\mu})\cdot\vec{e}_\nu\}^2$. The sum of the projection error and the variance of projection are constant. So maximizing on is equivalent to minimizing the other.

Also, this relationship can be expressed in terms of eigenvalues. It reduces to

$$\sum_{\nu=1}^{N}\lambda_\nu=\sum_{\nu=1}^{M}\lambda_\nu+\sum_{\nu=M+1}^{D}\lambda_\nu$$

To see this, $\frac{1}{N}\sum_{i=1}^{N}(\vec{x}_i-\vec{\mu})\cdot(\vec{x}_i-\vec{\mu})=\text{Trace}(\mathbf{C})=\sum_{\nu=1}^{D}\lambda_\nu$. The variance of the projection is $\sum_{\nu=1}^{M}\lambda_\nu$, by similar reasoning. Hence, the projection error is $\sum_{\nu=M+1}^{D}\lambda_\nu$.

1.4 Cost Function for PCA

The cost function for PCA can be defined as

$$J(\vec{M},\{a\},\{e\})=\sum_{k=1}^{N}||(\vec{\mu}+\sum_{i=1}^{M}a_{ki}\vec{e}_i)-\vec{x}_k||^2),$$

where The $\{a_{ki}\}$ are projection coefficients.

Minimize J w.r.t. $\vec{M},\{a\},\{e\}$ Data $\{\vec{x}_k : k=1 \text{ to N}\}$
Intuition: find the M-dimensional subspace s.t. the projections of the data onto this subspace have minimal error, see figure (5).

Minimizing J, gives the $\{\hat{\vec{e}}_i\}$'s to be the eigenvectors of the covariance matrix
$\vec{K}=\frac{1}{N}\sum_{k=1}^{N}(\vec{x}_k-\vec{\mu})(\vec{x}_k-\vec{\mu})^T$
$\vec{\mu}=\frac{1}{N}\sum_{k=1^N}\vec{x}_k$
$\hat{a}_{ki}=(\vec{x}_k-\hat{\vec{\mu}})\cdot\vec{e}_i$ the projection coefficients.

To fully understand why PCA minimizes or maximizes these terms we must express the criterion slightly differently. Then we use Singular Value Decomposition (SVD), which is advanced material of this lecture.

We can re-express the criteria as

$$J[\vec{\mu},\{a\},\{e\}]=\sum_{k=1}^{N}\sum_{b=1}^{D}\{(\mu_b-x_{bk})+\sum_{i=1}^{M}a_{ki}e_{ib})\}^2,$$

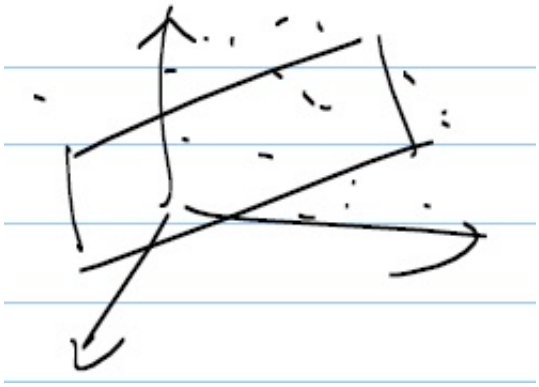

Figure 5: PCA can be obtained as the projection which minimizes the least square error of the residuals.

where b denotes the vector components.

This is an example of a general class of problem.

Let $E[\Psi, e] = \sum_{a=1,k=1}^{a=D,k=N} (\tilde{x}_{ak} - \sum_{\nu=1}^{M} \Psi_{a\nu}\Phi_{\nu k})^2$.

Goal: minimize $E[\Psi, e]$ w.r.t. Ψ, e.

This is a bilinear problem, that can be solved by SVD.

Note: $\tilde{x}_{ak} = x_{ak} - \mu_a$ is the position of the point, relative to the mean.

2 Singular Value Decomposition SVD

We can express any $N \times D$ matrix $\vec{X}$, x_{ak} in form

$$\mathbf{X} = \mathbf{E}\ \mathbf{D}\ \mathbf{F}$$

$$x_{ak} = \sum_{\mu,\nu=1}^{M} e_{a\mu} d_{\mu\nu} f_{\nu k}$$

where $\mathbf{D} = \{d_{\mu\nu}\}$ is a diagonal matrix ($d_{\mu\nu} = 0, \mu \neq \nu$). Note: $\mathbf{X}$ is not a square matrix (unless $D = N$). So it has no eigenvalues or eigenvectors.

$$\mathbf{D} = \begin{pmatrix} \sqrt{\lambda_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sqrt{\lambda_N} \end{pmatrix}, \text{ where the } \{\lambda_i\} \text{ are eigenvalues of } \mathbf{X}\ \mathbf{X}^T \text{(equivalently of } \mathbf{X}^T\ \mathbf{X})$$

$\mathbf{E} = \{e_{a\mu}\}$ are eigenvectors of $(\mathbf{X}\ \mathbf{X}^T)_{ab}$,

$\mathbf{F} = \{f_{\nu k}\}$ are eigenvectors of $(\mathbf{X}^T\ \mathbf{X})_{kl}$,

μ, ν label the eigenvectors.

Note: For $\bar{\mathbf{X}}$ defined on previous page, we get that $(\tilde{\mathbf{X}}\ \bar{\mathbf{X}}^T) = \sum_{k=1}^{N}(\vec{x}_k - \underline{\mu})(\vec{x}_k - \underline{\mu})^T$. Also note that if $(\mathbf{X}\ \mathbf{X}^T)\vec{e} = \lambda\vec{e}$, then $(\mathbf{X}^T\ \mathbf{X})(\mathbf{X}^T\vec{e}) = \lambda(\mathbf{X}^T\vec{e})$.

This relates the eigenvectors of $\mathbf{X}\ \mathbf{X}^T$ and of $\mathbf{X}^T\ \mathbf{X}$ (calculate the eigenvectors for the smallest matrix, then deduce those of the bigger matrix – usually $D < N$).

Minimize:

$$E[\psi, e] = \sum_{a=1,k=1}^{a=D,k=N} (\tilde{x}_{ak} - \sum_{\nu=1}^{M} \psi_{a\nu}\phi_{\nu k})^2$$

We set $\begin{cases} \psi_{a\nu} = \sqrt{\delta_{\nu\nu}}e_a^\nu \\ \phi_{\nu k} = \sqrt{\delta_{\nu\nu}}f_k^\nu \end{cases}$

Take M biggest terms in the SVD expansion of $\mathbf{x}$.
But there is an ambiguity.

$$\sum_{\nu=1}^{M} \psi_{a\nu}\phi_{\nu k} = (\psi\phi)_{ak} = (\psi\mathbf{A}\mathbf{A}^{-1}\phi)_{ak}$$

for any M $\times$ M invertible matrix $\mathbf{A}$
$\psi \to \psi\mathbf{A}$
$\phi \to \mathbf{A}^{-1}\phi$

For the PCA problem, we have constants that the projection directions one orthogonal unit eigenvectors. This gets rid of the ambiguity.

2.1 Relate SVD to PCA

Linear algebra can be used to relate SVD to PCA. Start with an $n \times m$ matrix $\mathbf{X}$.
$\mathbf{X}\mathbf{X}^T$ is a symmetric $n \times n$ matrix
$\mathbf{X}^T\mathbf{X}$ is a symmetric $m \times m$ matrix
Note that $(\mathbf{X}\mathbf{X}^T)^T = \mathbf{X}\mathbf{X}^T$.

By standard linear algebra,

$$\mathbf{X}\mathbf{X}^T\vec{e}^\mu = \lambda^\mu\vec{e}^\mu,$$

with n eigenvalues λ^μ and eigenvectors $\vec{e}^\mu$. The eigenvectors are orthogonal $\vec{e}^\mu \cdot \vec{e}^\nu = \delta^{\mu\nu}$ ($= 1$ if $\mu = \nu$, $= 0$ if $\mu \neq \nu$).

Similarly,

$$\mathbf{X}\mathbf{X}^T\vec{f}^\nu = \tau^\nu\vec{f}^\nu,$$

with m eigenvalues τ^ν and eigenvectors $\vec{f}^\nu$, where $\vec{f}^\mu \cdot \vec{f}^\nu = \delta^{\mu\nu}$.

The $\{\vec{e}^{\mu}\}$ and $\{\vec{f}^{\nu}\}$ are related because

$$(\mathbf{X}^T\mathbf{X})(\mathbf{X}^T\vec{e}^{\mu}) = \lambda^{\mu}(\mathbf{X}^T\vec{e}^{\mu})$$

$$(\mathbf{X}\mathbf{X}^T)(\mathbf{X}\vec{f}^{\mu}) = \tau^{\mu}(\mathbf{X}\vec{f}^{\mu})$$

Hence, $\mathbf{X}^T\vec{e}^{\mu} \propto \vec{f}^{\mu}$, $\mathbf{X}\vec{f}^{\mu} \propto \vec{e}^{\mu}$ and $\lambda^{\mu} = \tau^{\mu}$. If n >m , then there are n eigenvectors $\{\vec{e}_{\mu}\}$ and m eigenvectors $\{\vec{f}_{\mu}\}$. So, several $\{\vec{e}_{\mu}\}$ relate to the same $\vec{f}_{\mu}$.

Claim: we can express

$$\mathbf{X} = \sum_{\mu} \alpha^{\mu}\vec{e}^{\mu}\vec{f}_{\mu}^{T}$$

$$\mathbf{X}^T = \sum_{\mu} \alpha^{\mu}\vec{f}^{\mu}\vec{e}_{\mu}^{T}$$

(For some α^{μ}. We will solve for all α^{μ} later.)

Verify the claim:

$$\mathbf{X}\vec{f}^{\nu} = \sum_{\mu} \alpha^{\mu}\vec{e}^{\mu}\vec{f}_{\mu}^{T}\vec{f}_{\nu} = \sum_{\mu} \alpha^{\mu}\delta_{\mu\nu}\vec{e}^{\mu} = \alpha^{\nu}\vec{e}^{\nu}$$

$$\mathbf{X}\mathbf{X}^T = \sum_{\mu,\nu} \alpha^{\nu}\vec{e}^{\nu}\vec{f}_{\nu}^{T}\alpha^{\mu}\vec{f}^{\mu}\vec{e}_{\mu}^{T} = \sum_{\mu,\nu} \alpha^{\nu}\alpha^{\mu}\vec{e}^{\nu}\delta_{\mu\nu}\vec{e}_{\mu}^{T} = \sum_{\mu}(\alpha^{\mu})^2\vec{e}^{\mu}\vec{e}_{\mu}^{T}$$

Similarly, $\mathbf{x}^T\mathbf{x} = \sum_{\mu}(\alpha^{\mu})^2\vec{f}^{\mu}\vec{f}_{\mu}^{T}$. So, $(\alpha^{\mu})^2 = \lambda^{\mu}$. (Because we can express any symmetric matrix in form $\sum_{\mu}\lambda_{\mu}\vec{e}^{\mu}\vec{e}_{\mu}{}^{T}$, where λ^{μ} are the eigenvalues and $\vec{e}^{\mu}$ are eigenvectors.)

$\mathbf{X} = \sum_{\mu}\alpha^{\mu}\vec{e}^{\mu}\vec{f}_{\mu}{}^{T}$ is the SVD of $\mathbf{X}$

In coordinates:

$x_{ai} = \sum_{\mu}\alpha^{\mu}e_a^{\mu}f_i^{\mu}$

$x_{ai} = \sum_{\mu,\nu}e_a^{\mu}\alpha^{\mu}\delta_{\mu\nu}f_i^{\nu}$

$\mathbf{x} = \mathbf{EDF}$

$E_{a\mu} = e_a^{\mu}, \mathbf{D}_{\mu\nu} = \alpha^{\mu}\delta_{\mu\nu}, F_{\nu i} = f_i^{\nu}$.

3 Fisher's Linear Discriminant

PCA may not be the best way to reduce the dimension if the goal is discrimination. Suppose you want to discriminate between two classes of data 1&2, shown in figure (6).

If you put both sets of data into PCA, you will get this, see figure (7). The eigenvectors are $\vec{e}_1, \vec{e}_2$ with eigenvalues $\lambda_1 > \lambda_2$. Because of the form of the data $\lambda_1 >> \lambda_2$.

The best axis, according to PCA is in the worst direction for discrimination (best axis is $\vec{e}_1$ because $\lambda_1 >> \lambda_2$).

Projecting datasets onto $\vec{e}_1$ gives, see figure (3):

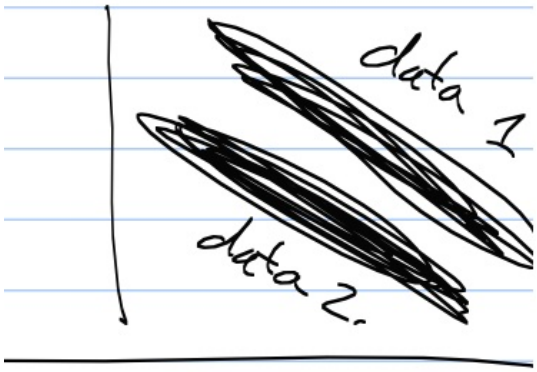

Figure 6: This type of data is bad for PCA. Fisher's Linear Discriminant does better of the goal is discrimination.

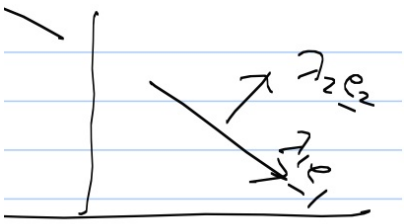

Figure 7: The PCA projections for the data in figure (6) The best axis, according to PCA, is the worst axis for projection if the goal is discrimination.

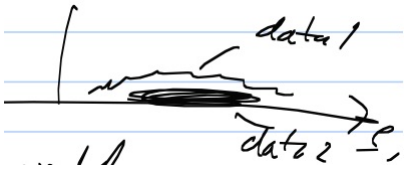

Figure 8: If we project the data onto $\vec{e}_1$, then data 1 and 2 gets all mixed together. Very bad for discrimination.

The second direction $\vec{e}_1$ would be for better. This would give, see figure (3):

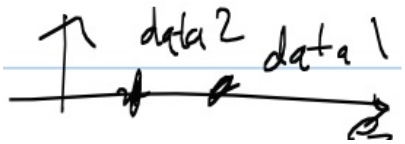

Figure 9: We get a better discrimination if we project the data onto $\vec{e}_2$, then it is easy to separate data 1 from data 2.

Fisher's Linear Discriminant gives a way to find a better projection direction.
n_1 samples $\vec{x}_i$ from class X_1
n_2 samples $\vec{x}_i$ from class X_2

The goal is to find a vector $\vec{w}$, project data onto this axis (i.e. $\vec{x}_i \cdot \vec{w}$) so that the data is well separated. Define the sample mean
$\vec{m}_i = \frac{1}{N_i} \sum_{\vec{x} \in X_i} \vec{x}$ for $i = 1, 2$.
Define the scatter matrices
$\mathbf{S}_i = \sum_{\vec{x} \in X_i} (\vec{x} - \vec{m}_i)(\vec{x} - \vec{m}_i)^T$ for $i = 1, 2$.
Define the between-class scatter
$\mathbf{S}_B = (\vec{x} - \vec{m}_i)(\vec{x} - \vec{m}_i)^T$ between classes X_1 and X_2.
Finally define the within-class scatter
$\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2$

Now, project onto the (unknown) direction $\vec{w}$
$\hat{m}_i = \frac{1}{N_i} \sum_{\vec{x} \in X_i} \vec{w} \cdot \vec{x} = \vec{x} \cdot \vec{m}_i$, using the definitions of the sample means. Note that the means of the projections are the projections of the means. The scatter of the projected points is
$\hat{S}_i^2 = \sum_{x \in X_i} (\vec{w} \cdot \vec{x} - \vec{w} \cdot \vec{m}_i)^2 = \vec{w}^T \, \mathbf{S}_i \, \vec{w}$, by definition of the scatter matrices.

The *Fisher criterion* is to choose the projection direction $\vec{w}$ to maximize:

$$J(\vec{w}) = \frac{|\hat{\vec{m}}_1 - \hat{\vec{m}}_2|^2}{\hat{S}_1^2 + \hat{S}_2^2}$$

This maximizes the ratio of the between-class distance ($|\hat{\vec{m}}_1 - \hat{\vec{m}}_2|$) to the within-class scatter.

See figure (10) with an example of a good projection direction, while other projection directions are bad – see figure (11).

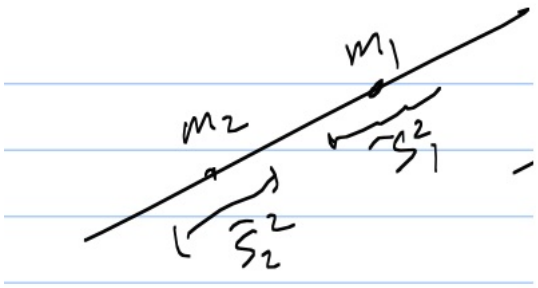

Figure 10: The projection of the data from figure (6) onto the best direction $\vec{w}$ (at roughly forty-five degrees). This separates the data well because the distance between the projected means m_1, m_2 is a lot bigger than the projected scatters $\hat{S}_1, \hat{S}_2$.

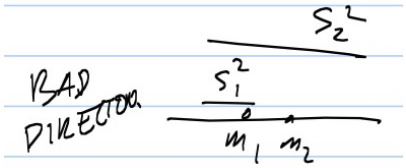

Figure 11: A bad projection direction, by comparison to figure (10). The distance between the projected means m_1, m_2 is smaller than the projected scatters $\hat{S}_1, \hat{S}_2$.

Result: The projection direction that maximizes $J(\vec{\omega})$ is $\vec{\omega} = \mathbf{S}_{\omega}^{-1}(\vec{m}_1 - \vec{m}_2)$. Note that this is not normalized (i.e. $|\vec{\omega}| \neq 1$), so we must normalize the vector.

Proof: Note that the Fisher criterion is independent of the norm of $\vec{\omega}$. So we can maximize it by arbitrarily requiring that $\vec{\omega}^T \mathbf{S}_{\omega} \vec{\omega} = \tau$, where τ is a constant. This can be formulated in term of maximization with constraints:

Maximize $\vec{\omega}^T \mathbf{S}_B \vec{\omega} - \lambda(\vec{\omega}^T \mathbf{S}_{\omega} \vec{\omega} - \tau)$, where λ is a Lagrange multiplier and τ is a constant.

$\frac{\delta}{\delta \vec{\omega}} \rightarrow \mathbf{S}_B \vec{\omega} - \lambda \mathbf{S}_{\omega} \vec{\omega} = 0$

Hence, $\mathbf{S}_{\omega}^{-1} \mathbf{S}_B \vec{\omega} = \lambda \vec{\omega}$. But
$\mathbf{S}_B = (\vec{m}_1 - \vec{m}_2)^T (\vec{m}_1 - \vec{m}_2)$
$\mathbf{S}_B \cdot \vec{\omega} = \rho(\vec{m}_1 - \vec{m}_2)$ for some ρ $(= \vec{w} \cdot (\vec{m}_1 - \vec{m}_2))$.

Hence $\mathbf{S}_B \hat{\vec{\omega}} \propto (\vec{m}_1 - \vec{m}_2)$. This implies that $\mathbf{S}_w^{-1} \mathbf{S}_B \hat{\vec{\omega}} \propto \mathbf{S}_w^{-1}(\vec{m}_1 - \vec{m}_2)$, and the result follows from recalling that $\mathbf{S}_{\omega}^{-1} \mathbf{S}_B \vec{\omega} = \lambda \vec{\omega}$.

3.1 Alternative

An alternative way to model this problem is to assign a Gaussian model to each dataset (i.e. learn the model parameters $\vec{\mu}, \mathbf{\Sigma}$ for each dataset). Then if the *covariance is the same for both datasets*, then the Bayes classifier is a straight line whose normal is the direction $\vec{\omega}$.

$\vec{\omega} \cdot \vec{x} + \omega_0 = 0, \vec{\omega} = \mathbf{\Sigma}(\vec{\mu}_1 - \vec{\mu}_2)$. This is exactly the same as Fisher's method! See figure (12).

But if the data comes from two Gaussian with different covariances, then Bayes classifier is a quadratic curve, so it differs from Fisher's linear discriminant.

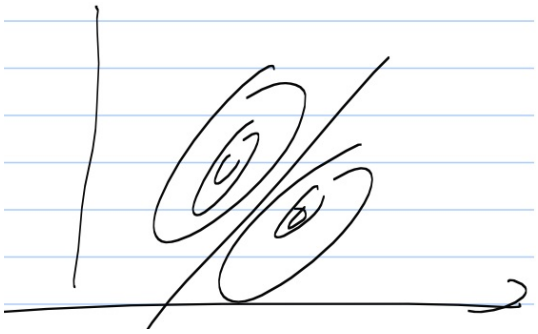

Figure 12: Try to model the datasets by learning Gaussian models for each dataset separately. Then we recover Fisher if both datasets have the same covariance. The decision plane will have a surface normal which points along the direction of Fisher's $\vec{\omega}$. But if the two datasets have different covariances, then the Bayes classifier will be a quadratic curve and differs from Fisher.

3.2 Multiple Classes

For c classes, compute $c-1$ discriminants project D-dimensional feature with $c-1$ space.

Within-class

$\mathbf{S}_\omega = \mathbf{S}_1 + \cdots + \mathbf{S}_{c-1}$

Between-class

$\mathbf{S}_B = \mathbf{S}_{total} - \mathbf{S}_\omega = \sum_{i=1}^{c} n_i \cdot (\vec{m}_i - \vec{m})(\vec{m}_i - \vec{m})^T$, where $\mathbf{S}_{total}$ is the scatter matrix for all the classes.

Multiple Discriminant Analysis consists of the following steps:

Seek vectors $\omega_i : i = 1, \ldots, c-1$

Project samples to $c-1$ dim space: $(\omega_1 \cdot x, \ldots, \omega_{c-1} \cdot x) = \omega^T \vec{x}$.

The criterion is $J(\omega) = \frac{|\omega^T \mathbf{S}_B \omega|}{|\omega^T \mathbf{S}_\omega \omega|}$, where$|.|$ is the determinant.

The solution is given by the eigenvectors, where eigenvalues are the $c-1$ largest in $\mathbf{S}_B \vec{W} = \lambda \mathbf{S}_w \vec{w}$

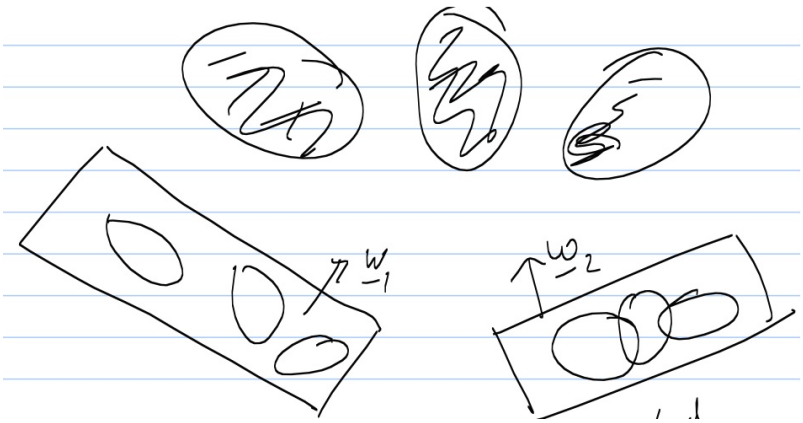

Figure 13: w^1 is a good project direction, w^2 is a bad projection direction.

$\vec{w}^1$ is a good projection of the data, $\vec{w}_2$ is a bad projection, see figure (13).

3.3 Limitations of PCA and Fisher

It is important to realize the limitations of these methods and also why they are popular.

They are popular party because they are easy to implement. They both have optimization criteria which can be solved by linear algebra. This is because the optimization criteria are quadratic, so the solutions are linear. This restriction was necessary when computers did not exist.

But now it is possible to have other optimization criteria which can be solved by computers. For example, based on criteria like nearest neighbour classification. This will be discussed later.

Also PCA assumes that the data lies on a low-dimensional linear space. But what if it lies on a low-dimensional curved space? More advanced techniques can deal with this – e.g. ISOMAP.