Linear Algebra Methods for Data Mining

Saara Hyvönen, Saara.Hyvonen@cs.helsinki.fi

Spring 2007

Linear Discriminant Analysis

Principal components analysis

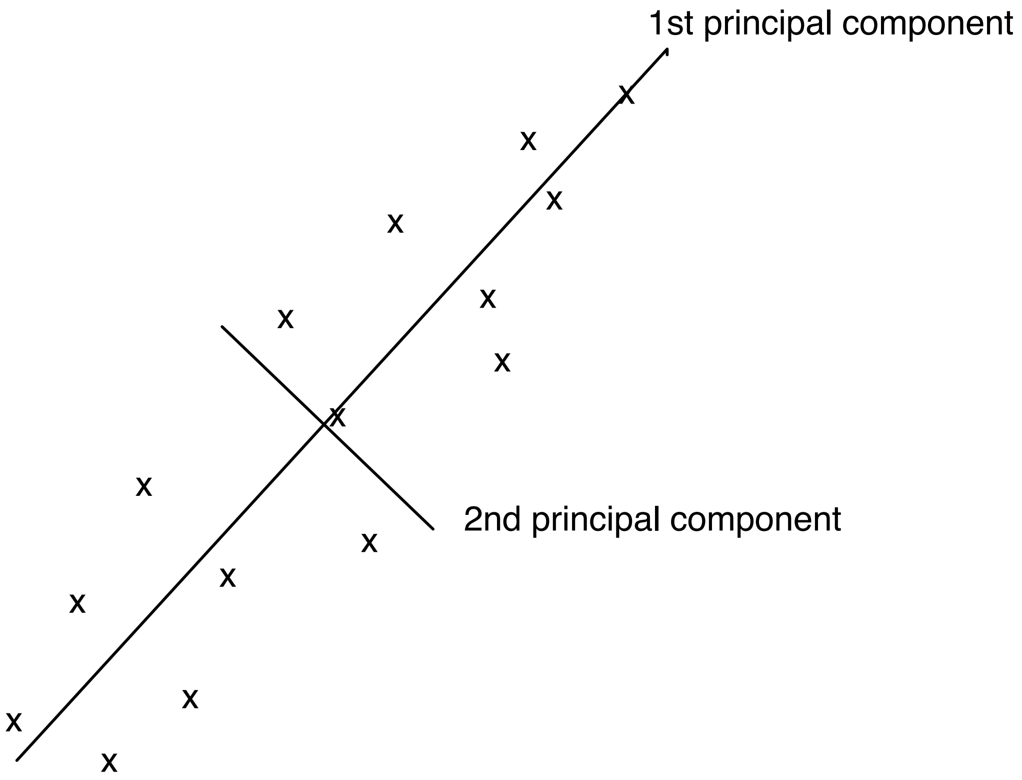
- Idea: look for such a direction that the data projected onto it has maximal variance.

- When found, continue by seeking the next direction, which is orthogonal to this (i.e. uncorrelated), and which explains as much of the remaining variance in the data as possible.

- Ergo: we are seeking linear combinations of the original variables.

- If we are lucky, we can find a few such linear combinations, or directions, or (principal) components, which describe the data fairly accurately.

- The aim is to capture the intrinsic variability in the data.


1st principal component
2nd principal component

How to compute the PCA:

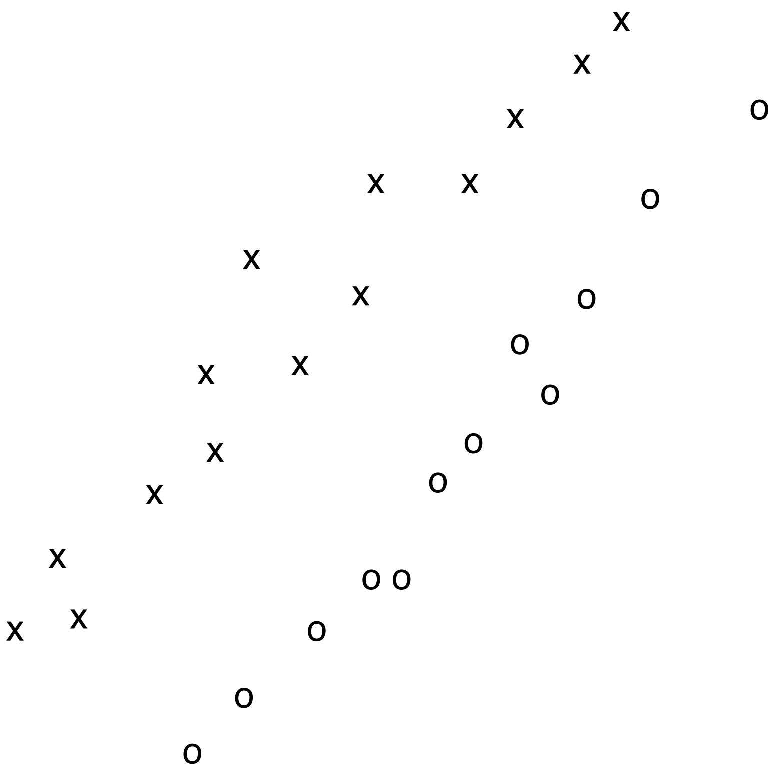
Data matrix $\mathbf{A}$, rows=data points, columns = variables (attributes, parameters).

1. Center the data by subtracting the mean of each column.

2. Compute the SVD of the centered matrix $\hat{\mathbf{A}}$ (or the k first singular values and vectors):
$$\hat{\mathbf{A}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T.$$

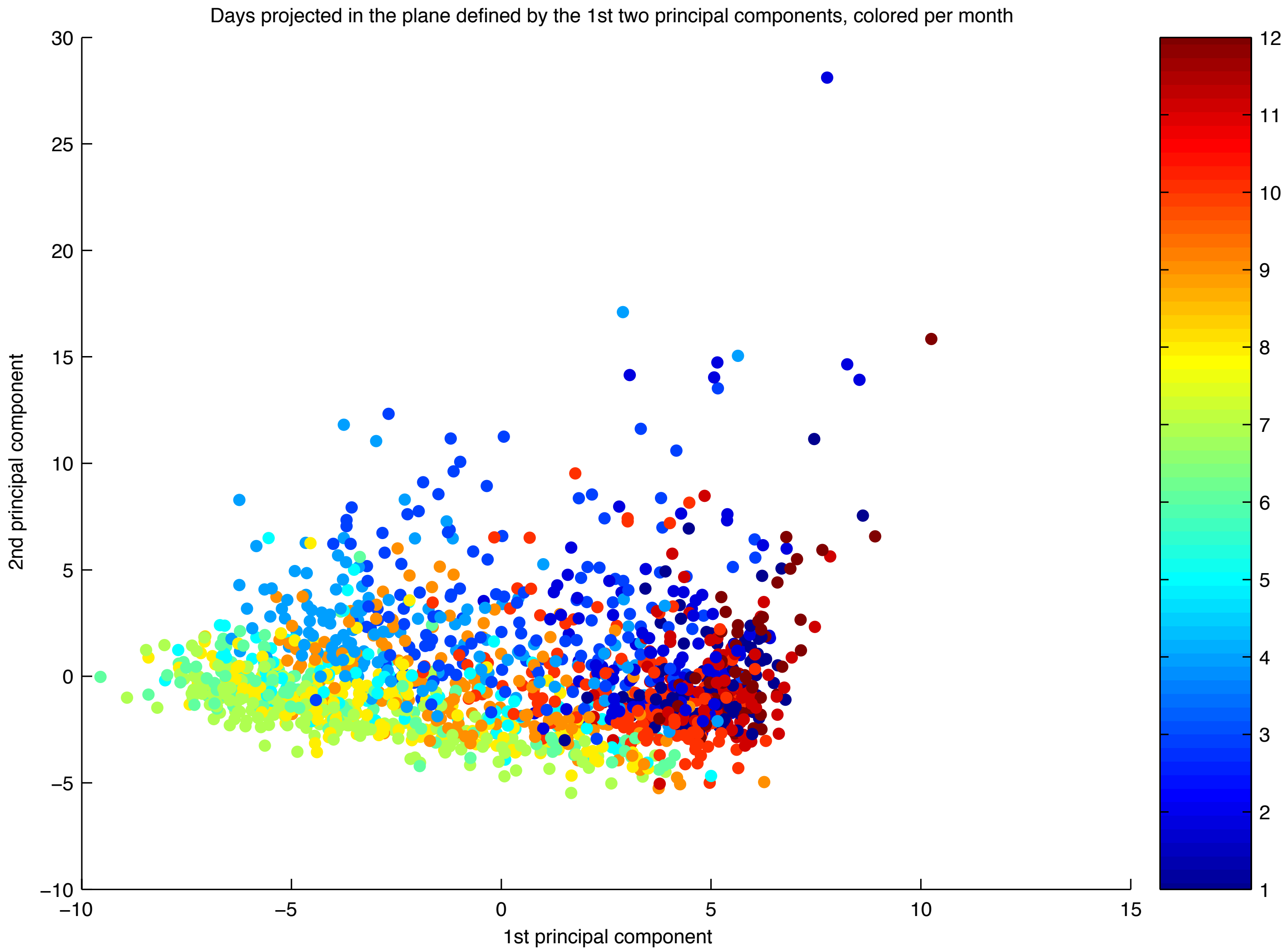
3. The principal components are the columns of $\mathbf{V}$, the coordinates of the data in the basis defined by the principal components are $\mathbf{U}\mathbf{\Sigma}$.

But the PC's are not always what we want!



Example: Atmospheric data

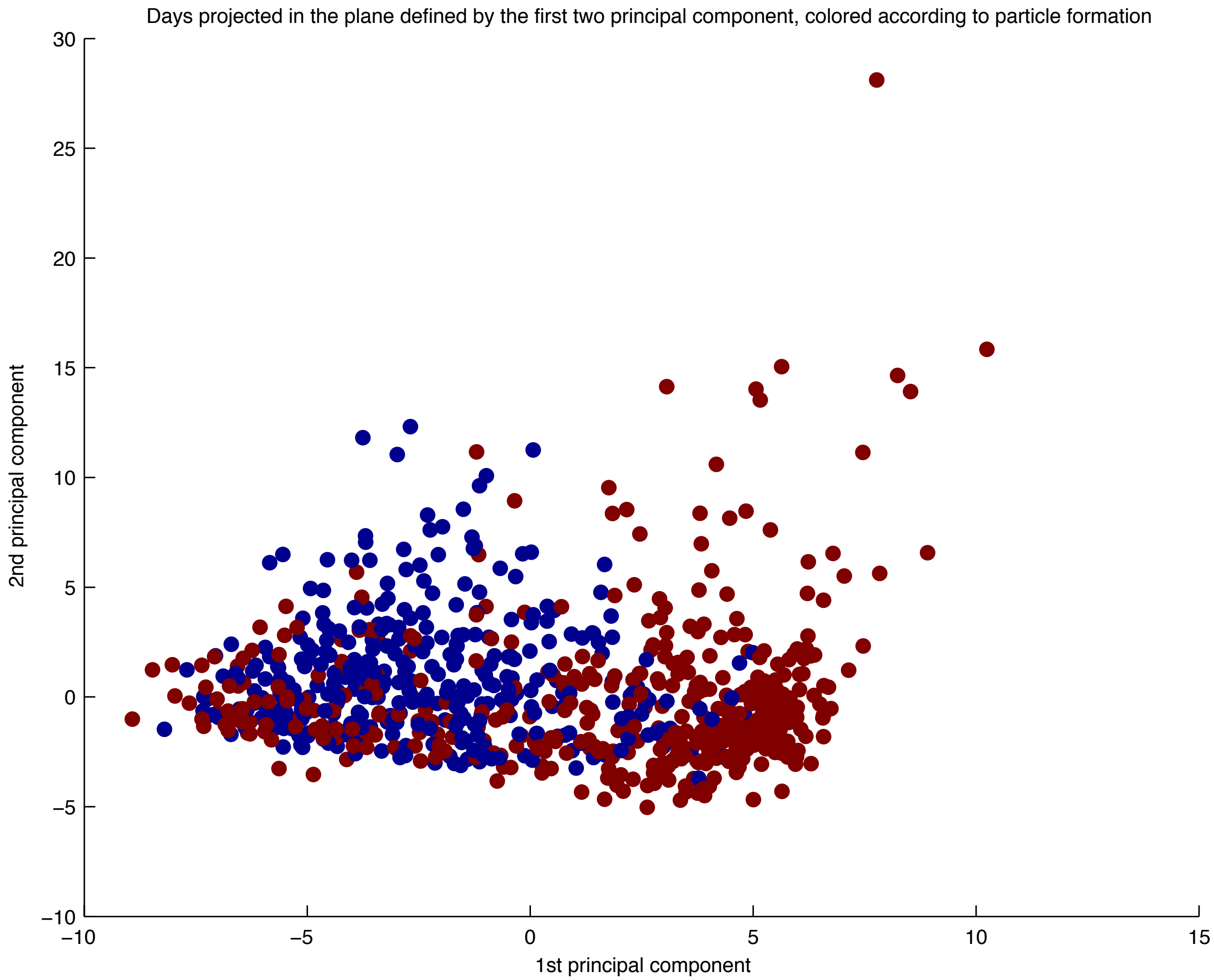
- Data: 1500 days, and for each day, we have the means and stds of around 30 measured variables (temperature, wind speed and direction, rain fall, UV-A radiation, concentration of CO2 etc.)
- Therefore, our data matrix is 1500×60.
- Visualizing things in a 60-dimensional space is challenging!
- Instead, do PCA, and project days onto the plane defined by the first two principal components.


Days projected in the plane defined by the 1st two principal components, colored per month
2nd principal component
1st principal component
30
25
20
15
10
5
0
−5
−10
−10
−5
0
5
10
15
12
11
10
9
8
7
6
5
4
3
2
1

- But this is not really what we are interested in!

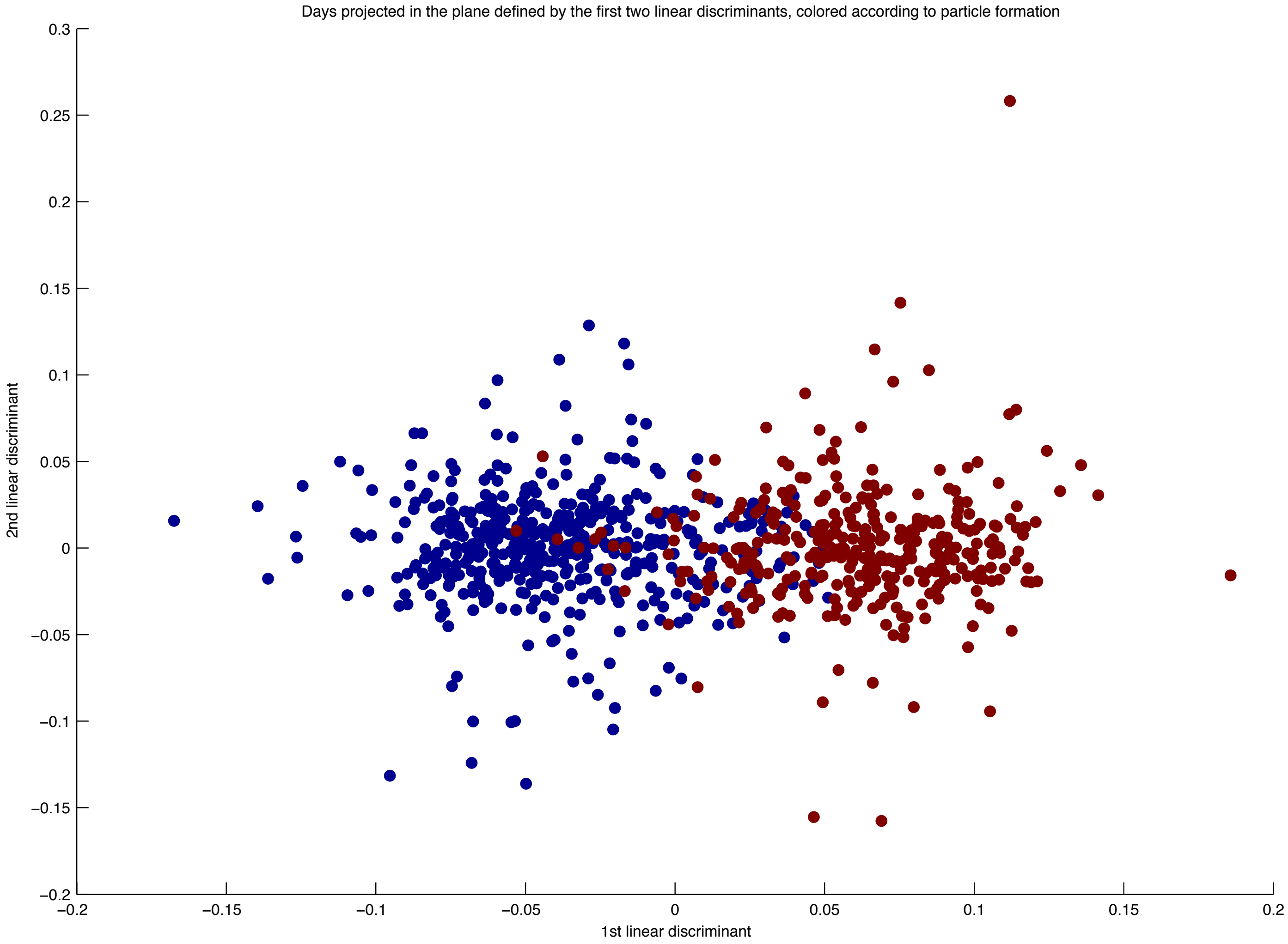
- Instead, we are interested in distinguishing days when new particles spontaneously form from days with no such formation.

- Prinicplal components are not very good at this!


Days projected in the plane defined by the first two principal component, colored according to particle formation
30
25
20
15
10
5
0
−5
−10
2nd principal component
−10
−5
0
5
10
15
1st principal component

What to do?

- Look instead for a direction, which
  - Minimized within-group variance
  - Maximized between-group variance.
- Project the data onto this direction: groups (should be) well separated!
- This is what Linear Discriminant Analysis does.


Days projected in the plane defined by the first two linear discriminants, colored according to particle formation
0.3
0.25
0.2
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
−0.2
2nd linear discriminant
−0.2
−0.15
−0.1
−0.05
0
0.05
0.1
0.15
0.2
1st linear discriminant

Linear Discriminant Analysis

- We are given the data matrix together with class labels: each data point belongs to one of the classes $1 \ldots k$.

- Goal: map the original data into features that most effectively discriminate between classes.

- In other words, reduce dimension of data in a way that best preserves its cluster structure.

Assume the columns of $\mathbf{A} \in \mathbb{R}^{m \times n}$ are grouped into k clusters:

$$\mathbf{A} = [\mathbf{A}_1 \ \mathbf{A}_2 \ \ldots \ \mathbf{A}_k], \quad \mathbf{A}_i \in \mathbb{R}^{m \times n_i}, \quad \sum_{i=1}^{k} n_i = n.$$

The centroid of each cluster ins computed by taking the average of the columns in $\mathbf{A}_i$:

$$\mathbf{c}_i = \frac{1}{n_i} \mathbf{A}_i \mathbf{e}^i, \quad \mathbf{e}^i = (1, \ \ldots \ 1)^T \in \mathbb{R}^{n_i \times 1},$$

and the global centroid is defined as

$$\mathbf{c} = \frac{1}{n} \mathbf{A}\mathbf{e}, \quad \mathbf{e} = (1, \ \ldots \ 1)^T \in \mathbb{R}^{n \times 1}.$$

Let N_i denote the set of column indices that belong to cluster $\mathbf{A}_i$. Then the within-cluster, between-cluster and mixture (or total) scatter matrices are defined as follows:

$$\mathbf{S}_w = \sum_{i=1}^{k} \sum_{j \in N_i} (\mathbf{a}_j - \mathbf{c}_i)(\mathbf{a}_j - \mathbf{c}_i)^T$$

$$\mathbf{S}_b = \sum_{i=1}^{k} \sum_{j \in N_i} (\mathbf{c}_i - \mathbf{c})(\mathbf{c}_i - \mathbf{c})^T = \sum_{i=1}^{k} n_i (\mathbf{c}_i - \mathbf{c})(\mathbf{c}_i - \mathbf{c})^T$$

$$\mathbf{S}_m = \sum_{i=1}^{n} (\mathbf{a}_j - \mathbf{c})(\mathbf{a}_j - \mathbf{c})^T$$

Let $\mathbf{w}$ be the vector along which we shall project our data.

Now we achieve our goal by maximizing the objective:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

In doing so, we minimize the within-cluster scatter while maximizing the between-cluster scatter.

Note: one can show that $\mathbf{S}_m = \mathbf{S}_w + \mathbf{S}_b$, so

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

can be written as

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_m \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}} - 1$$

which means we are maximizing total scatter while minimizing within-cluster scatter.

Note, that the value of

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

is the same regardless of how we scale $\mathbf{w} \rightarrow \alpha \mathbf{w}$.

Before (when discussing PCA) we chose $\mathbf{w}^T \mathbf{w} = 1$.

This time, let us require $\mathbf{w}$ to be such that $\mathbf{w}^T \mathbf{S}_w \mathbf{w} = 1$.

Now our problem can be stated as follows: we wish to maximize

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

subject to the constraint $\mathbf{w}^T \mathbf{S}_w \mathbf{w} = 1$.

Optimization problem: maximize

$$f = \mathbf{w}^T \mathbf{S}_b \mathbf{w} - \lambda(\mathbf{w}^T \mathbf{S}_w \mathbf{w} - 1),$$

λ is the Lagrange multiplier.

Again we solve the optimization problem

$$\max_{w} f = \max_{w} \left(\mathbf{w}^T \mathbf{S}_b \mathbf{w} - \lambda (\mathbf{w}^T \mathbf{S}_w \mathbf{w} - 1) \right)$$

by differentiating with respect to $\mathbf{w}$; this yields

$$\frac{\partial f}{\partial \mathbf{w}} = 2\mathbf{S}_b \mathbf{w} - 2\lambda \mathbf{S}_w \mathbf{w} = 0$$

This leads to the *generalized eigenvalue problem*

$$\mathbf{S}_b \mathbf{w} = \lambda \mathbf{S}_w \mathbf{w}.$$

If $\mathbf{S}_w$ is invertible, the generalized eigenproblem

$$\mathbf{S}_b\mathbf{w} = \lambda \mathbf{S}_w\mathbf{w}.$$

can be written as

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w} = \lambda\mathbf{w}.$$

The solutions of this are the eigenvalues and eigenvectors of the matrix $\mathbf{S}_w^{-1}\mathbf{S}_b$.

Denote the eigenvalues and eigenvectors by λ_k and $\mathbf{w}_k$.

Remembering, that

$$\mathbf{S}_b \mathbf{w}_k = \lambda_k \mathbf{S}_w \mathbf{w}_k$$

insert these into $J(\mathbf{w})$:

$$J(\mathbf{w}_k) = \frac{\mathbf{w}_k^T \mathbf{S}_b \mathbf{w}_k}{\mathbf{w}_k^T \mathbf{S}_w \mathbf{w}_k} = \frac{\mathbf{w}_k^T \lambda_k \mathbf{S}_w \mathbf{w}_k}{\mathbf{w}_k^T \mathbf{S}_w \mathbf{w}_k} = \lambda_k.$$

So the direction $\mathbf{w}$ which maximizes the value of $J(\mathbf{w})$ is the eigenvector corresponding to the largest eigenvalue of $\mathbf{S}_w^{-1}\mathbf{S}_b$.

The largest eigenvalue tells about how well classes separate.

Generalized eigenvalue problems

$\mathbf{A}$, $\mathbf{B} \in \mathbb{R}^{n \times n}$.

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{B}\mathbf{x}, \quad \mathbf{x} \neq \mathbf{0}.$$

Has n generalized eigenvalues λ if and only if rank$\mathbf{B} = n$.

If rank$\mathbf{B} < n$, then the number of λ may be zero, finite, or infinite:

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ 0 & 3 \end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix} 1 & 2 \\ 0 & 3 \end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix},$$

$$\mathbf{B} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}. \quad \mathbf{B} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}. \quad \mathbf{B} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}.$$

Symmetric-definite generalized eigenproblems

Let the matrices $\mathbf{A}$, $\mathbf{B} \in \mathbb{R}^{n \times n}$, be such that $\mathbf{A}$ symmetric, $\mathbf{B}$ symmetric positive definite.

Find λ and $\mathbf{x} \neq \mathbf{0}$ such that

$$\mathbf{Ax} = \lambda \mathbf{Bx}.$$

Theorem. If $\mathbf{A}$, $\mathbf{B} \in \mathbb{R}^{n \times n}$, $\mathbf{A}$ symmetric, $\mathbf{B}$ symmetric positive definite, then there exists a nonsingular $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ such that

$$\mathbf{X}^T\mathbf{A}\mathbf{X} = \text{diag}(a_1, \ldots, a_n), \quad \mathbf{X}^T\mathbf{B}\mathbf{X} = \text{diag}(b_1, \ldots, b_n).$$

Moreover, $\mathbf{A}\mathbf{x}_i = \lambda_i\mathbf{B}\mathbf{x}_i$ for $i = 1, \ldots, n$, where $\lambda_i = a_i/b_i$.

Note: the matrix $\mathbf{X}$ can be chosen in such a way, that

$$\mathbf{X}^T\mathbf{A}\mathbf{X} = \text{diag}(\lambda_1, \ldots, \lambda_n), \quad \mathbf{X}^T\mathbf{B}\mathbf{X} = \text{diag}(1, \ldots, 1).$$

Example

$$\mathbf{A} = \begin{pmatrix} 229 & 163 \\ 163 & 116 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 81 & 59 \\ 59 & 43 \end{pmatrix}$$

$$\mathbf{X} = \begin{pmatrix} 3 & -5 \\ -4 & 7 \end{pmatrix}$$

Our generalized eigenvalue problem was

$$\mathbf{S}_b\mathbf{w} = \lambda\mathbf{S}_w\mathbf{w}.$$

We know that both $\mathbf{S}_b$ and $\mathbf{S}_w$ are symmetric, and positive semidefinite. Assuming $\mathbf{S}_w$ is invertible it is also positive definite. So there is a matrix $\mathbf{X}$ such that

$$\mathbf{X}^T\mathbf{S}_b\mathbf{X} = \text{diag}(\lambda_1, \ldots, \lambda_n), \quad \mathbf{X}^T\mathbf{S}_w\mathbf{X} = \text{diag}(1, \ldots, 1),$$

and

$$\mathbf{S}_b\mathbf{x}_i = \lambda_j\mathbf{S}_w\mathbf{x}_i.$$

Since $\mathbf{S}_b$ is positive semidefinite, and $\mathbf{x}_i^T\mathbf{S}_b\mathbf{x}_i = \lambda_j$, we see that the $\lambda_i \geq 0$ for all i.

So, the generalized eigenvalues of our problem

$$\mathbf{S}_b \mathbf{w} = \lambda \mathbf{S}_w \mathbf{w}$$

are all nonnegative.

Moreover, only the largest r eigenvalues are nonzero, where $r = \text{rank}(\mathbf{S}_b)$.

Remember that

$$\mathbf{S}_b = \sum_{i=1}^{k} n_i (\mathbf{c}_i - \mathbf{c})(\mathbf{c}_i - \mathbf{c})^T,$$

which is a sum of k rank-1 matrices, so the rank of $\mathbf{S}_b$ is at most k.

Here we only considered looking for the first linear discriminant, which is the eigenvector corresponding to the largest eigenvalue of $\mathbf{S}_w^{-1}\mathbf{S}_b$.

The l first linear discriminants are (of course!) the eigenvectors corresponding to the l largest eigenvalues.

So *how* did we get the linear discriminants?

Step 1: Compute the scatter matrices $\mathbf{S}_b$ and $\mathbf{S}_w$.

Step 2: Solve the generalized eigenvalue problem $\mathbf{S}_b\mathbf{w} = \lambda\mathbf{S}_w\mathbf{w}$.
(In matlab you can use eigs!)

Step 3: Order the eigenvalues from largest to smallest,
and the eigenvectors accordingly. These are your linear discriminants.

Step 4: In classification: use training set to decide where boundaries are. Use test set to evaluate performance.

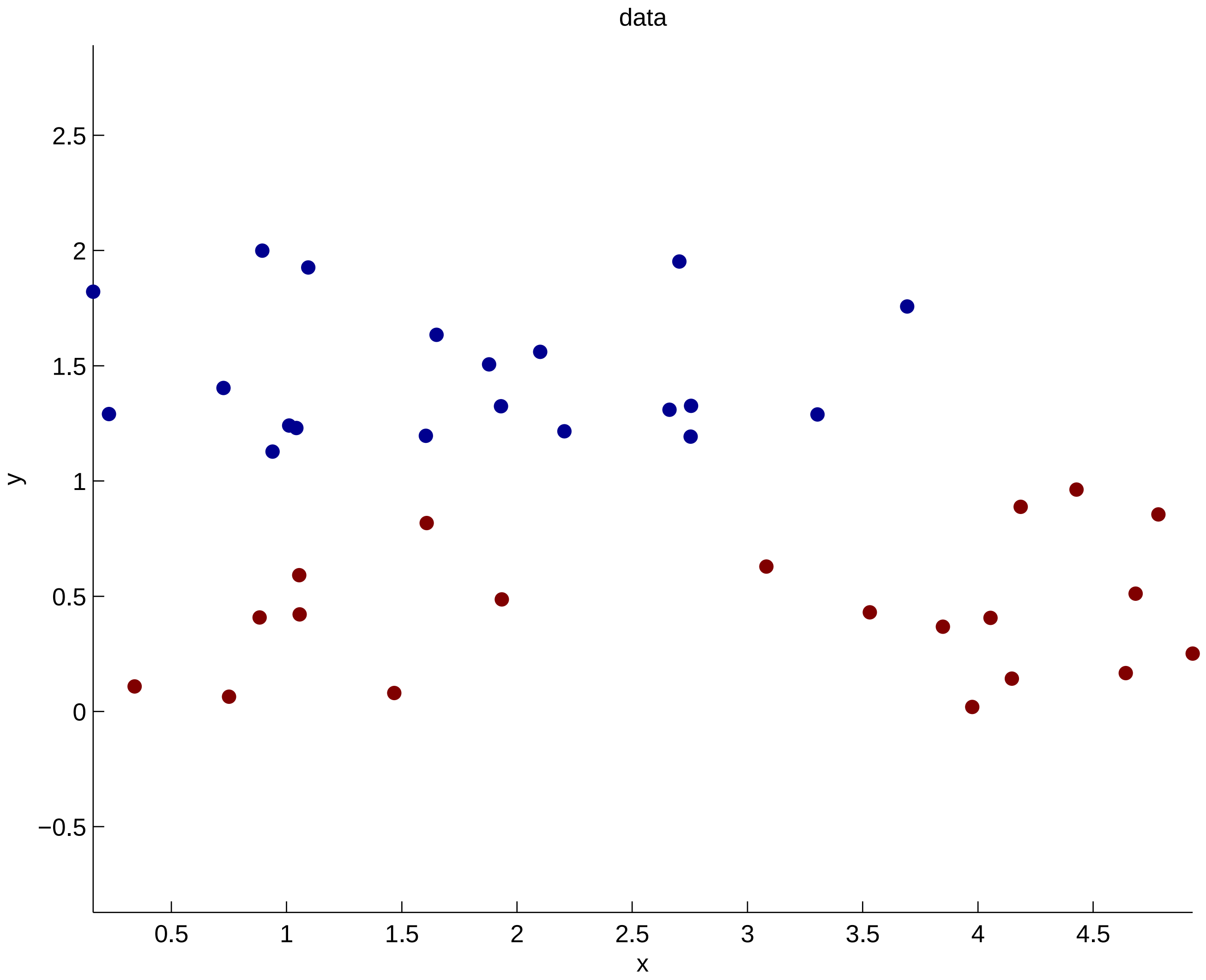
Note: in reality, one should use N-fold crossvalidation:

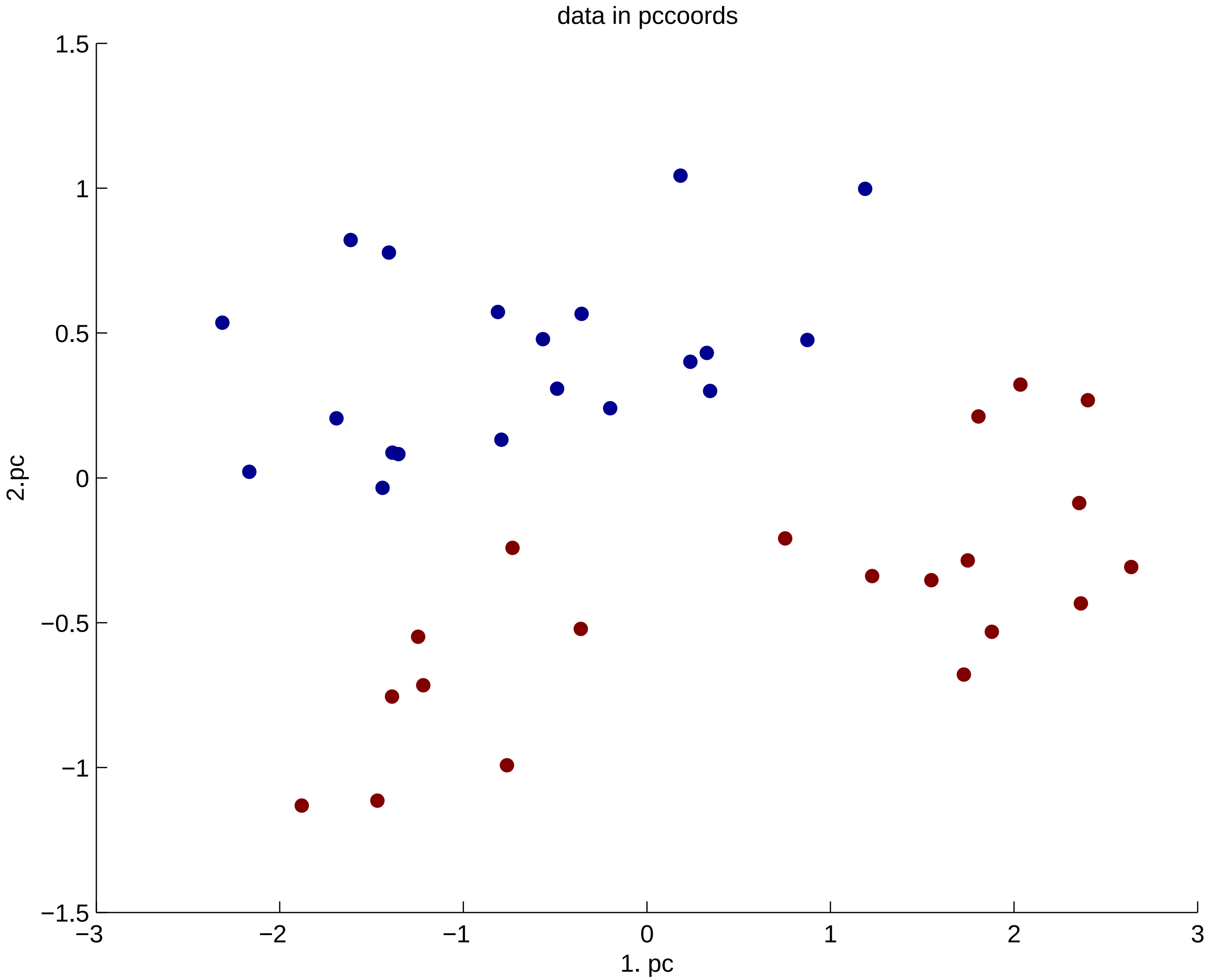
Divide all data into N parts, and use one part as the test set and the rest as the training set. Report the average performance across the test sets.

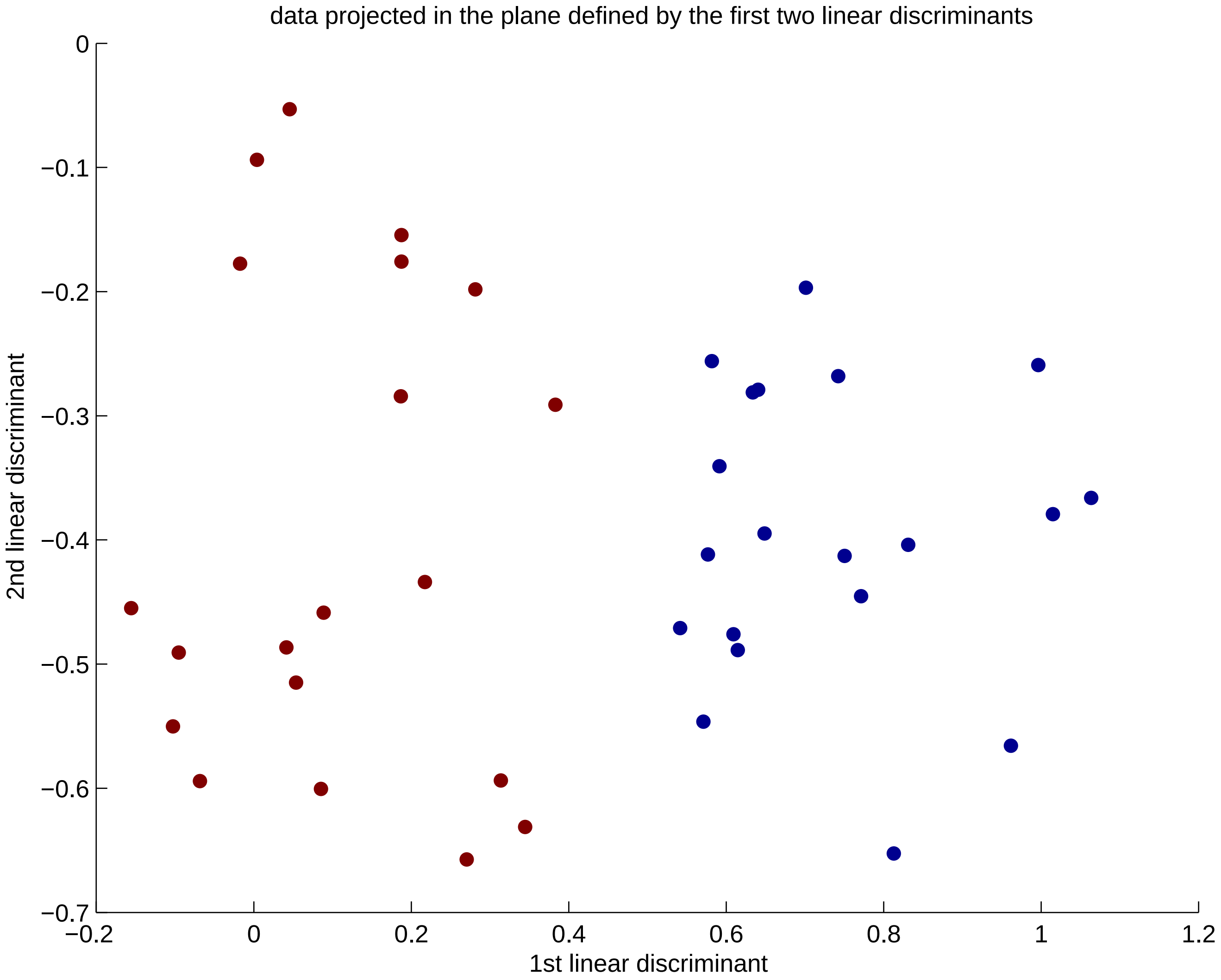
Why? To get a more reliable estimate of performance (avoiding the situation where the training set and test set are ”too good”).

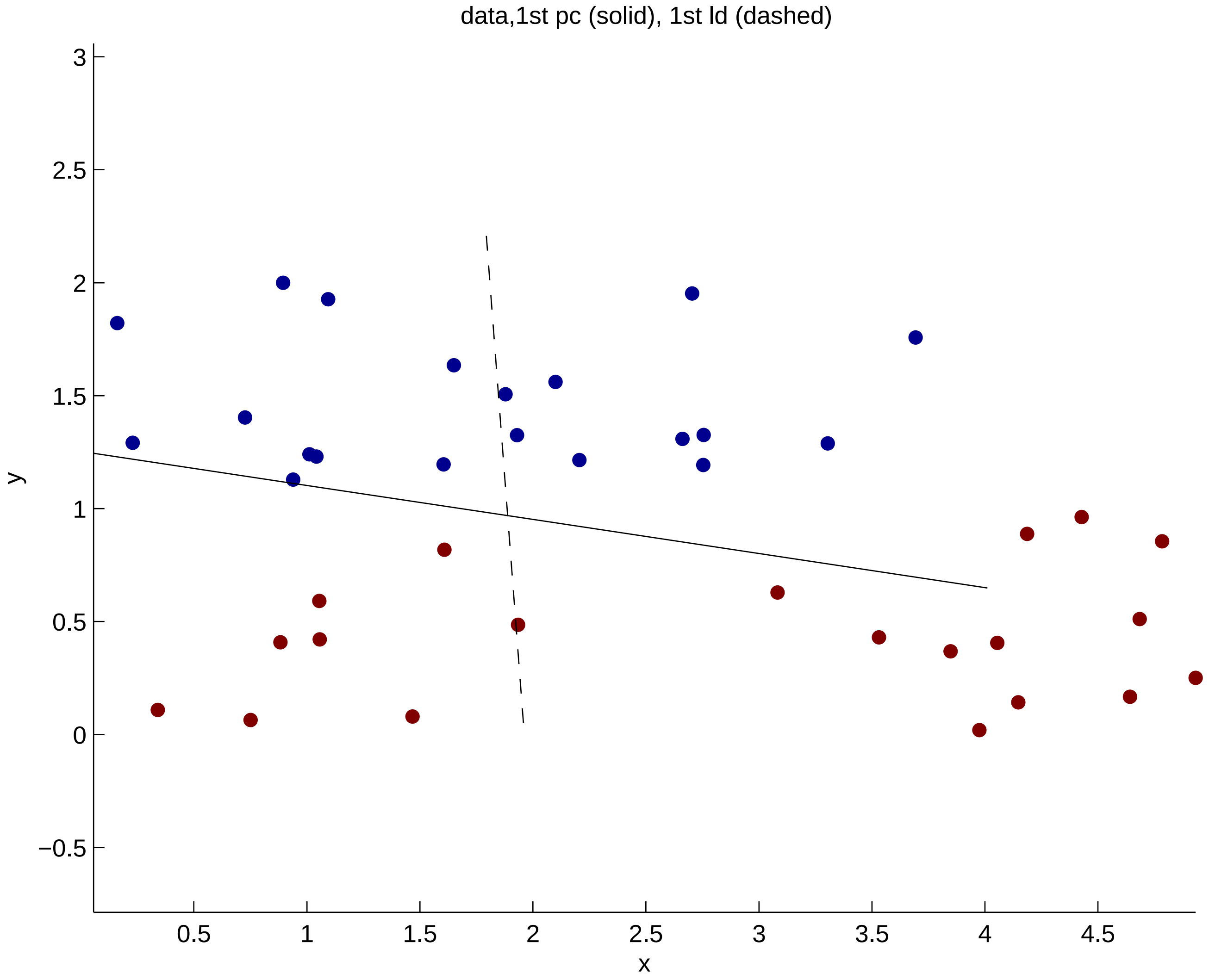
Example: 2 classes in 2D

```
c1=mean(A1,2);c2=mean(A2,2);
A=[A1 A2];c=mean(A,2);
sb=n1*(c1-c)*(c1-c)'+n2*(c2-c)*(c2-c)';
tmp1=A1-repmat(c1,1,n1);
tmp2=A2-repmat(c2,1,n2);
sw=tmp1*tmp1'+tmp2*tmp2';
[v,d]=eigs(sb,sw);
```


data
y
x
2.5
2
1.5
1
0.5
0
−0.5
0.5
1
1.5
2
2.5
3
3.5
4
4.5


data in pccoords
1.5
1
0.5
0
−0.5
−1
−1.5
2.pc
−3
−2
−1
0
1
2
3
1. pc


data projected in the plane defined by the first two linear discriminants
2nd linear discriminant
1st linear discriminant
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.2
0
0.2
0.4
0.6
0.8
1
1.2


data,1st pc (solid), 1st ld (dashed)
y
x
3
2.5
2
1.5
1
0.5
0
−0.5
0.5
1
1.5
2
2.5
3
3.5
4
4.5

Two-class case

$$\mathbf{S}_w = \sum_{j \in N_1} (\mathbf{a}_j - \mathbf{c}_1)(\mathbf{a}_j - \mathbf{c}_1)^T + \sum_{j \in N_2} (\mathbf{a}_j - \mathbf{c}_2)(\mathbf{a}_j - \mathbf{c}_2)^T = n_1 \mathbf{\Sigma}_1 + n_2 \mathbf{\Sigma}_2,$$

where $\mathbf{\Sigma}_i$ is the covariance matrix of class i.

Also, we can use the fact that $\mathbf{c} = \frac{1}{n}(n_1 \mathbf{c}_1 + n_2 \mathbf{c}_2)$ to get

$$\mathbf{S}_b = n_1(\mathbf{c}_1 - \mathbf{c})(\mathbf{c}_1 - \mathbf{c})^T + n_2(\mathbf{c}_2 - \mathbf{c})(\mathbf{c}_2 - \mathbf{c})^T = \frac{n_1 n_2}{n}(\mathbf{c}_2 - \mathbf{c}_1)(\mathbf{c}_2 - \mathbf{c}_1)^T.$$

This is a rank-1 matrix!

We have, for nonsingular $\mathbf{S}_w$,

$$\mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{x}_1 = \mathbf{S}_w\frac{n_1n_2}{n}(\mathbf{c}_2-\mathbf{c}_1)(\mathbf{c}_2-\mathbf{c}_1)^T\mathbf{x}_1 = \lambda_1\mathbf{x}_1,$$

which yields (for some α)

$$\mathbf{x}_1 = \alpha\mathbf{S}_w^{-1}(\mathbf{c}_2-\mathbf{c}_1),$$

and

$$\lambda_1 = \frac{n_1n_2}{n}(\mathbf{c}_2-\mathbf{c}_1)^T\mathbf{S}_w^{-1}(\mathbf{c}_2-\mathbf{c}_1)\big(= \mathrm{trace}(\mathbf{S}_w^{-1}\mathbf{S}_b)\big).$$

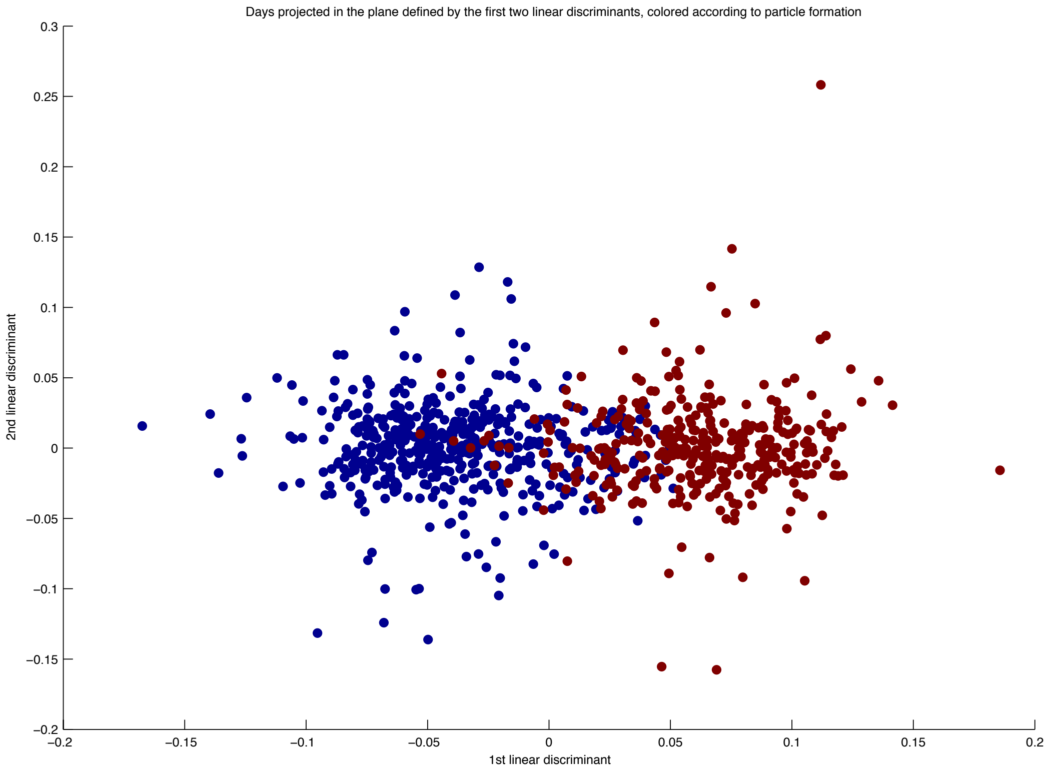
Good class separation?

Remember: the largest eigenvalue tells about how well classes separate.

$$\lambda_1 = \frac{n_1 n_2}{n}(\mathbf{c}_2 - \mathbf{c}_1)^T \mathbf{S}_w^{-1}(\mathbf{c}_2 - \mathbf{c}_1)$$

So we get better separation of two classes, if difference of class means $(\mathbf{c}_2 - \mathbf{c}_1)$ is large relative to the weighted sum of class covariance matrices $n_1 \mathbf{\Sigma}_1 + n_2 \mathbf{\Sigma}_2 = \mathbf{S}_w$.

Atmospheric data again



Could we look at the weights of the variables in the first linear discriminant to see which variables are important in separating the red dots from the blue?

Fisher discriminant analysis

(R.A. Fisher, The use of multiple measurements in taxonomic problems, 1936)

Uses slighlty different criterion: maximize

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

where the scatter matrices are

$$\mathbf{S}_b = (\mathbf{c}_2 - \mathbf{c}_1)(\mathbf{c}_2 - \mathbf{c}_1)^T, \quad \mathbf{S}_w = \mathbf{\Sigma}_1 + \mathbf{\Sigma}_2.$$

If the classes are of equal size ($n_1 = n_2$), then this is the same as what we discussed above.

Other criteria

Several measures of cluster quality, which involve the three scatter matrices, have been suggested, including

$$J = \mathsf{trace}(\mathbf{S}_w^{-1}\mathbf{S}_b)$$

and

$$J = \mathsf{trace}(\mathbf{S}_w^{-1}\mathbf{S}_m).$$

For more discussion on these and others, see e.g. [2] and references therein.

What if S_w is singular?

- Then this approach will not work, as it is based on finding the eigenvalues of $\mathbf{S}_w^{-1}\mathbf{S}_b$!

- This is typically the case in undersampled problems, where the number of samples is small compared to the dimension of the data points.

- For example, microarray data, text data, image data.

- Answer: instead of solving the generalized eigenproblem we can formulate the problem in terms of the generalized SVD.

References

[1] Lars Eldén: Matrix Methods in Data Mining and Pattern Recognition, SIAM 2007.

[2] P. Howland and H. Park: Extension of Discriminant Analysis based on the Generalized Singular Value Decomposition, 2002.

[3] B. G. Ripley, Pattern Recognition and Neural Networks, Cambridge University Press, 1996.