# AIC-5102B Natural Language Processing

## Additional lecture notes on machine learning

X. Hilaire

ESIEE Paris, Départment IT

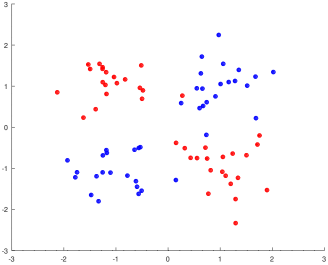
22 septembre 2023

# Aims

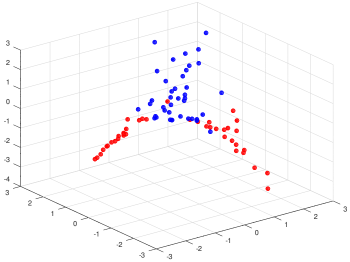
- To introduce two classification methods, that will prove useful for the NLP task of sentiment analysis, and the assessed lab :
  - Linear discriminant analysis (LDA)
  - Support Vector Classification (SVC)
- Both methods have been introduced to find linear decision boundaries.
- But extensions to the non-linear case will be presented too thanks to the kernel trick.

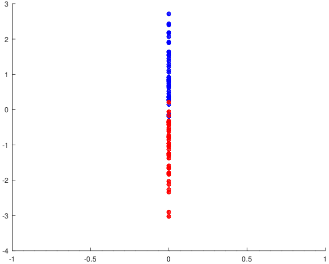
# The kernel trick

- Some data may not be separable in their original space, but may become separable by embedding them in a higher dimensional space
- Example 1

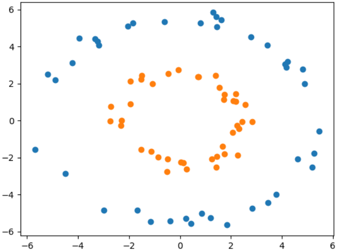


- Applying the transformation $\phi : (x, y) \mapsto (x, y, xy)$, then projecting the result onto the $Oz$ axis :

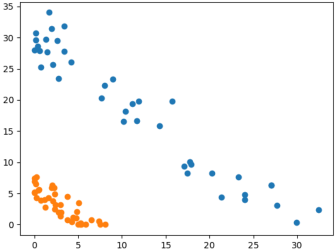




- Example 2



By applying the transformation $\phi : (x, y) \mapsto (x^2, y^2)$



**Lemma**

*If some data are linearly separable in a vector space $\mathbb{E}$, then they are also linearly separable in any superspace of $\mathbb{E}$.*

*Proof.* Let $d$ be the dimension of $\mathbb{E}$. Let $\boldsymbol{x}_i \in \mathbb{E}, i = 1, ..., m$ and $\boldsymbol{x}_j \in \mathbb{E}, j = 1, ..., n$ also be the data to separate. If these data are linearly separable, there must exists $\boldsymbol{w} \in \mathbb{E}$ and $b$ such that

$$\boldsymbol{w}\boldsymbol{x}_i + b \geq 0 \forall i = 1, ..., m$$
$$\boldsymbol{w}\boldsymbol{y}_j + b < 0 \forall j = 1, ..., n$$

Now let $\mathbb{E}'$ be a superspace of $\mathbb{E}$, obtained by adding $d' - d$ dimensions to $\mathbb{E}$. Then the above inequalities are unchanged by adding $d' - d$ zeros to $\boldsymbol{w}$, i.e. by choosing $\boldsymbol{w}' = (\boldsymbol{w}, \mathbf{0})$, and $b' = b$.□

Consequences :

- By increasing the dimensionality of the classification space, we increase the chances that the data in it will be separable
- Learning data very often requires finding a vector space large enough for the data to be separable there after transformation.
- It is our interest to increase the dimensionality of the classification space only by its necessary amount : beyond that, the risk is to overfit
- However, the necessary dimensionality can increase very quickly

Q : Do we have to explicitly construct the transformed data in high-dimensional space ?
A : No, as long as the classification algorithm does not use anything else than dot products

The kernel trick :

- If an algorithm only requires to compute dot products $< \boldsymbol{x}, \boldsymbol{y} >$ between pairs of points $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{E}$, then switching to $\mathbb{E}'$ will only require computing $< \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) >$
- If these quantities are computable **directly** from $\boldsymbol{x}$ and $\boldsymbol{y}$, and not from their images $\phi(\boldsymbol{x})$, $\phi(\boldsymbol{y})$, then the explicit calculation of their images is useless
- By choosing what the scalar product must equal to in the high-dimensional space, we implicitly induce a space, as well as the representation of the data in that space, but we never have to explicitly represent data in that space : this is known as **the kernel trick**

Illustration : our two previous examples involve either the squares or the products of the components of the data to be separated.

Consider $K(\boldsymbol{x},\boldsymbol{y}) = (1+ <\boldsymbol{x},\boldsymbol{y}>)^2$. Then,

$$\begin{aligned}
K(\boldsymbol{x},\boldsymbol{y}) &= (1+ <\boldsymbol{x},\boldsymbol{y}>)^2 \\
&= 1 + 2 <\boldsymbol{x},\boldsymbol{y}> + <\boldsymbol{x},\boldsymbol{y}>^2 \\
&= 1 + 2\boldsymbol{x}_1\boldsymbol{y}_1 + 2\boldsymbol{x}_2\boldsymbol{y}_2 + \boldsymbol{x}_1^2\boldsymbol{y}_1^2 + \boldsymbol{x}_2^2\boldsymbol{y}_2^2 + 2\boldsymbol{x}_1\boldsymbol{y}_1\boldsymbol{x}_2\boldsymbol{y}_2 \\
&= [1, \sqrt{2}\boldsymbol{x}_1, \sqrt{2}\boldsymbol{x}_2, \boldsymbol{x}_1^2, \boldsymbol{x}_2^2, \sqrt{2}\boldsymbol{x}_1\boldsymbol{x}_2] \\
&\quad [1, \sqrt{2}\boldsymbol{y}_1, \sqrt{2}\boldsymbol{y}_2, \boldsymbol{y}_1^2, \boldsymbol{y}_2^2, \sqrt{2}\boldsymbol{y}_1\boldsymbol{y}_2]^t \\
&= \phi(\boldsymbol{x})^t\phi(\boldsymbol{y}) \\
&=< \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) >
\end{aligned}$$

- The function $\phi(\boldsymbol{x}) = [1, \sqrt{2}\boldsymbol{x}_1, \sqrt{2}\boldsymbol{x}_2, \boldsymbol{x}_1^2, \boldsymbol{x}_2^2, \sqrt{2}\boldsymbol{x}_1\boldsymbol{x}_2]^t$ induces a dot product in a space of dimension 6, 2 components of which carry the squares, and 1 the rectangle product.

- The knowledge of $K$ allows us to calculate the distance between any pair of points in high-dimensional space :

$$\begin{aligned}||\phi(\boldsymbol{x})-\phi(\boldsymbol{y})|| &= \sqrt{<\phi(\boldsymbol{x})-\phi(\boldsymbol{y}),\phi(\boldsymbol{x})-\phi(\boldsymbol{y})>}\\ &= \sqrt{K(\boldsymbol{x}-\boldsymbol{y},\boldsymbol{x}-\boldsymbol{y})}\end{aligned}$$

- Note that we started from the dot product to end up with $\phi$ in the previous example. Doing the converse, namely starting from $\phi$ to end up with $K$, is always easy. But factorizing, as we did it in the above example, is not always obvious.
- A function $K$ which is factorizable in that way is called a kernel.
- A function is a kernel if and only if it is positive semidefinite (consequence of Mercer's theorem)

### Definition

Let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric function. Then $K$ is a **positive definite kernel** if and only if, for any finite sequence $(x_1, c_1)$, $(x_2, c_2)$, ..., $(x_n, c_n) \in \mathcal{X} \times \mathbb{R}$, the following holds true :

$$\sum_{i=1}^{n} \sum_{j=1}^{n} K(x_i, x_j) c_i c_j \geq 0$$

Remark : this is equivalent to saying that the **Gram matrix**, that is, the matrix whose generic term is $K(x_i, x_j)$, is positive semidefinite.

Let $\mathcal{X}$ be a compact set, and $k : \mathcal{X}^2 \to \mathbb{R}$ be a symmetric function. Consider the integral operator $T_k : L_2(\mathcal{X}) \to L_2(\mathcal{X})$ [1], defined as

$$T_k f(.) = \int_{\mathcal{X}} k(., x) f(x) dx \tag{1}$$

$T_k$ is said to be positive semidefinite if, for all $f \in L_2(\mathcal{X})$, we have that

$$\int_{\mathcal{X}^2} k(u, v) f(u) f(v) dudv \geq 0 \tag{2}$$

or, equivalently,

$$< f, T_k f >_{L_2(\mathcal{X})} \quad \geq 0$$

[1] $L_2(.)$ here denotes the set of square integrable functions over some compact set

### Theorem (Mercer)

*Let $k : \mathcal{X}^2 \to \mathbb{R}$ and $T_k$ be defined as in the last slide. Then, there exists a set of eigenfunctions $(\phi_i)_i \in L_2(\mathcal{X})$ with corresponding eigenvalues $\lambda_i \geq 0$ such as*

$$k(x, y) = \sum_{i=1}^{\infty} \lambda_i \phi_i(x) \phi_i(y)$$

*holds true for any $x, y \in \mathcal{X}$. Moreover, the convergence is absolute and uniform.*

- Mercer's theorem is stated in the continuous case, but its discrete version is easily established
- We first look for a discrete version of $T_k$, which can be obtained by replacing $f$ by a vector $(f_1, f_2, ..., f_d)$ in (1)

- Then $T_k$ becomes

$$T_k f(.) = \sum_{i=1}^{d} k(., x_i) f_i$$

- Condition (2) now writes

$$f^T K f \geq 0$$

where $K_{ij} = k(x_i, x_j)$, which means that $K$ is positive semidefinite, and can be written as

$$K = \sum_{t=1}^{d} \lambda_t e_t e_t^T$$

- Then,

$$k(x_i, x_j) = \left( \sum_{t=1}^{d} \lambda_t e_t e_t^T \right)_{ij}$$
$$= \sum_{t=1}^{d} \lambda_t (e_t)_i (e_t)_j$$
$$= \sum_{t=1}^{d} \lambda_t \phi_t(x_i) \phi_t(x_j)$$

where $\phi_t$ is chosen such as $\phi_t(x_i) = (e_t)_i$ for all $i$

## Corollary

*Let $\mathcal{X}$ be a compact set, and $k : \mathcal{X}^2 \to \mathbb{R}$ be a symmetric and continuous function. The following are equivalent :*

- *$k$ is a positive semidefinite kernel*
- *There exists eigenfunctions $(\phi_t)_t$ and related positive eigenvalues $(\lambda)_i$ such as*

$$k(x, y) = \sum_t \lambda_t \phi_t(x) \phi_t(y)$$

- *Every Gram matrix is positive semidefinite*
- *The integral operator $T_k$ is positive semidefinite.*

- Some commonly used generic kernels :
    - The inhomogeneous polynomial kernel of order $d$ : $K(\boldsymbol{x}, \boldsymbol{y}) = (1 + <\boldsymbol{x}, \boldsymbol{y}>)^d$
    - The Gaussian kernel : $K(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\frac{||\boldsymbol{x}-\boldsymbol{y}||^2}{2\sigma^2})$
    - The radial core : $K(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma||\boldsymbol{x} - \boldsymbol{y}||^2)$
    - The hyperbolic kernel : $K(\boldsymbol{x}, \boldsymbol{y}) = \tanh(\gamma <\boldsymbol{x}, \boldsymbol{y}> + \alpha)$
- The kernel trick can be applied to any algorithm, as long as it does not use anything more than dot dot products on the data
- This is the case of the LDA, and of SVM, of which we will now examine the nonlinear extensions.

# Fisher's discriminant analysis - LDA

Linear discriminant analysis

# Fisher's discriminant analysis - LDA

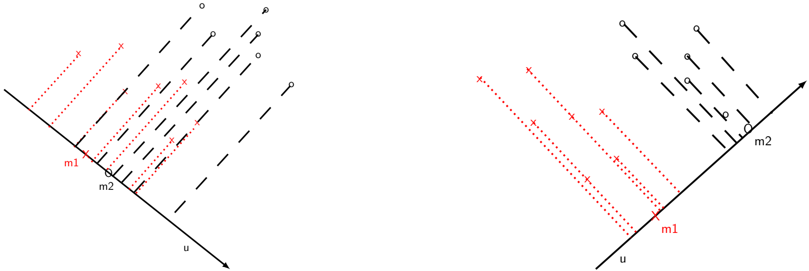
Motivations :

- We have data $\boldsymbol{X}$, in matrix form
- The data is separated into 2 classes $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, which are known
- We want to project the data on a single axis, i.e. $\boldsymbol{P}$ such that

$$\boldsymbol{x}' = \boldsymbol{P}\boldsymbol{X}$$

  be of dimension 1.
- How to choose $\boldsymbol{P}$ which ensures the best possible separation between classes ?

Drawing :

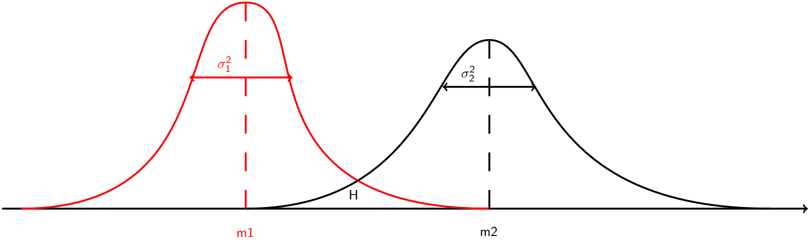


The projection of the figure on the right is clearly better :

- the distances between projected averages are greater
- there is less variance in each projected class

Fisher's discriminant idea : two classes separate well if

- Their intraclass variance is low
- Their interclass variance is high

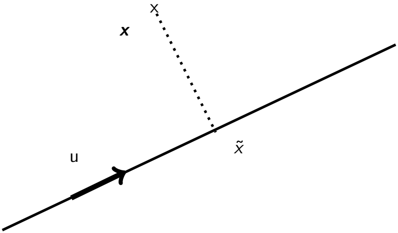


Fisher proposes to evaluate the quality of a separation by combining both variances in a ratio (sometimes called the Rayleigh ratio) :

$$J = \frac{\text{Variance interclasse}}{\text{Variance intraclasse}} = \frac{|m_1 - m_2|^2}{\sigma_1^2 + \sigma_2^2}$$

A ratio of $J = \infty$ indicates an ideal separation (and implies that the intraclass variance is zero). A ratio of $J = 0$ indicates a failed separation (and implies that the means are identical).

Idea of LDA (linear discriminant analysis) : look for a direction $\boldsymbol{u}$ which maximizes $J(\boldsymbol{u})$.



- A high-dimensional point $\boldsymbol{x}$ projects onto $\boldsymbol{u}$ at a point whose abscissa is $\tilde{x}$
- So, for class 1 : $E[\tilde{x}_1] = E[\boldsymbol{u} \cdot \boldsymbol{x}_1] = \boldsymbol{u} \cdot E[\boldsymbol{x}_1] = \boldsymbol{u} \cdot \boldsymbol{m}_1$
- And for class 2 : $E[\tilde{x}_2] = \boldsymbol{u} \cdot \boldsymbol{m}_2$

and the interclass variance is written

$$\begin{aligned} |E[\tilde{x}_1] - E[\tilde{x}_2]|^2 &= (\boldsymbol{u}^t(\boldsymbol{m}_1 - \boldsymbol{m}_2))^2 \\ &= \boldsymbol{u}^t(\boldsymbol{m}_1 - \boldsymbol{m}_2)(\boldsymbol{m}_1 - \boldsymbol{m}_2)^t\boldsymbol{u} \\ &= \boldsymbol{u}^t\boldsymbol{B}\boldsymbol{u} \end{aligned}$$

The intraclass variance of class 1, on the other hand, equals to

$$\begin{aligned} \mathrm{Var}[\tilde{x}_1] &= E[(\tilde{x}_1 - E[\tilde{x}_1])^2] \\ &= E[(\boldsymbol{x}_1 \cdot \boldsymbol{u} - \boldsymbol{m}_1\boldsymbol{u})^2] = E[((\boldsymbol{x}_1 - \boldsymbol{m}_1) \cdot \boldsymbol{u})^2] \\ &= E[\boldsymbol{u}^t(\boldsymbol{x}_1 - \boldsymbol{m}_1)(\boldsymbol{x}_1 - \boldsymbol{m}_1)^t\boldsymbol{u}] \\ &= \boldsymbol{u}^tE[(\boldsymbol{x}_1 - \boldsymbol{m}_1)(\boldsymbol{x}_1 - \boldsymbol{m}_1)^t]\boldsymbol{u} \\ &= \boldsymbol{u}^t\boldsymbol{S}_1\boldsymbol{u} \end{aligned}$$

where $\boldsymbol{S}_1$ is the variance-covariance matrix of $\boldsymbol{x}_1$ alone. For class 2, we obtain in the same way

$$\mathrm{Var}[\tilde{x}_2] = \boldsymbol{u}^t\boldsymbol{S}_2\boldsymbol{u}$$

The total intraclass variance is therefore equal to

$$\begin{aligned}\mathrm{Var}[\tilde{x}_1] + \mathrm{Var}[\tilde{x}_2] &= \boldsymbol{u}^t(\boldsymbol{S}_1 + \boldsymbol{S}_2)\boldsymbol{u} \\ &= \boldsymbol{u}^t\boldsymbol{W}\boldsymbol{u}\end{aligned}$$

where $\boldsymbol{W}$ is the intraclass variance-covariance matrix. The ratio to be maximized is therefore written, as a function of $\boldsymbol{u}$

$$J(u) = \frac{\boldsymbol{u}^t\boldsymbol{B}\boldsymbol{u}}{\boldsymbol{u}^t\boldsymbol{W}\boldsymbol{u}} \tag{3}$$

Note that it is scale invariant : $J(\alpha\boldsymbol{u}) = J(\boldsymbol{u})$ for $\alpha \neq 0$. So, provided that $\boldsymbol{W}$ is regular[2], maximizing $J(\boldsymbol{u})$ amounts to maximizing $\boldsymbol{u}^t\boldsymbol{B}\boldsymbol{u}$ under constraint that $\boldsymbol{u}^t\boldsymbol{W}\boldsymbol{u} = 1$.

[2] $\det(W) \neq 0$

We must therefore solve the following problem :

$$\begin{aligned} \underset{\boldsymbol{u}}{\min!} \quad & \boldsymbol{u}^t \boldsymbol{B} \boldsymbol{u} \\ \text{s.t.} \quad & ||\boldsymbol{u}^t \boldsymbol{W} \boldsymbol{u}|| = 1 \end{aligned}$$

The Lagrangian of the problem equals to

$$L(\lambda) = \boldsymbol{u}^t \boldsymbol{B} \boldsymbol{u} - \lambda(\boldsymbol{u}^t \boldsymbol{W} \boldsymbol{u} - 1)$$

We vanish its gradient in $\boldsymbol{u}$ :

$$\begin{aligned} \nabla_{\boldsymbol{u}} L = 2\boldsymbol{B}\boldsymbol{u} - 2\lambda \boldsymbol{W}\boldsymbol{u} &= 0 \\ \boldsymbol{W}^{-1}\boldsymbol{B}\boldsymbol{u} &= \lambda \boldsymbol{u} \end{aligned}$$

which shows that $\boldsymbol{u}$ must be an eigenvector of $\boldsymbol{W}\boldsymbol{B}$. It corresponds to a $> 0$ eigenvalue, because $\boldsymbol{W}^{-1}\boldsymbol{B}$ is positive definite. Solving for the characteristic polynomial yields

$$\boldsymbol{u} = \boldsymbol{W}^{-1}(\boldsymbol{m}_1 - \boldsymbol{m}_2) \tag{4}$$

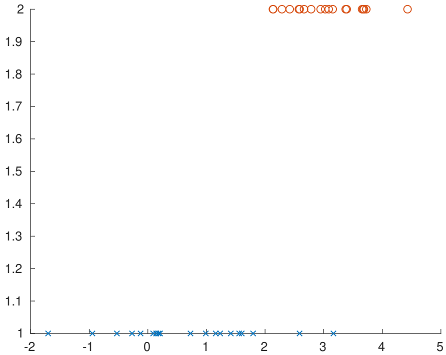
When the points of the two classes follow a multivariate Gaussian distribution with variance-covariance matrix $\boldsymbol{\Sigma}$ and expectations $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, then (4) is written

$$\boldsymbol{u} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

The projections of the variances of $\boldsymbol{\mu}_1$ and of $\boldsymbol{\mu}_2$ on $\boldsymbol{u}$ are then sufficient to decide whether a test point should be classified in class 1 or 2.

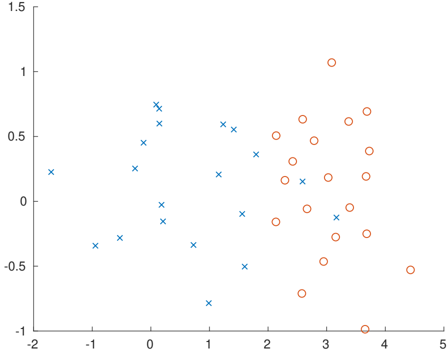
## Elementary example :

We generate 2 univariate Gaussian random variables ($X1$ plotted at $y = 1$, $X2$ at $y = 2$) :



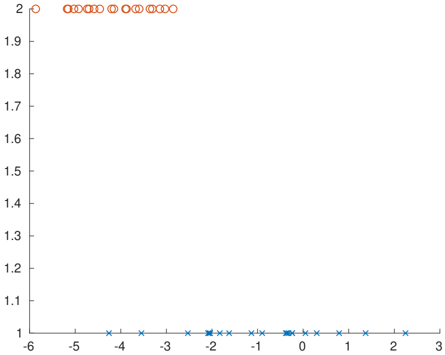
```
X1=mvnrnd([1],[1],20);
X2=mvnrnd([3],[0.5],20);
figure, scatter(X1, ones(20,1), 'x');
hold on;
scatter(X2, 2*ones(20,1), 'o');
```

We increase dimensionality :



```
U1=[X1, mvnrnd(0, 0.3, 20)];
U2=[X2, mvnrnd(0, 0.3, 20)];
figure, scatter(U1(:,1), U1(:,2), 'x');
hold on;
scatter(U2(:,1), U2(:,2), 'o');
```

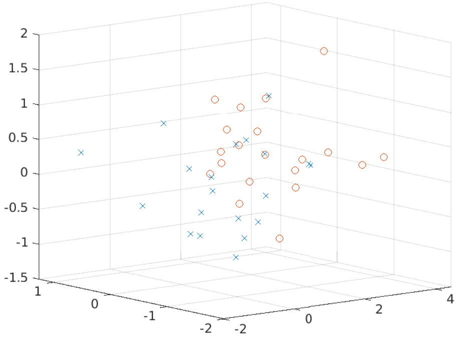
## What the LDA Finds :



```
% medium
m1=mean(U1);
m2=mean(U2);

% intraclass variance matrix
S1= 1/19*(U1-m1)'*(U1-m1);
S2=1/19*(U2-m2)'*(U2-m2);
W=S1+S2;
```

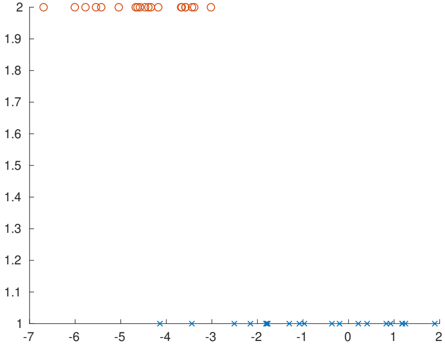
```
% solution
u= W^(-1)*(m1-m2)';

% reprojections
y1= U1*u;
y2= U2*u;
figure, scatter(y1, ones(20,1), 'x');
hold on;
scatter(y2, 2*ones(20,1), 'o');
```

## Same thing with 3 dimensions :



```
U1=[X1, mvnrnd(0, 0.3, 20),  mvnrnd(0, 0.3, 20)];
U2=[X2, mvnrnd(0, 0.3, 20),  mvnrnd(0, 0.3, 20)];
figure, scatter3(U1(:,1), U1(:,2), U1(:,3), 'x');
hold on;
scatter3(U2(:,1), U2(:,2), U2(:,3), 'o');
```

# What the LDA Finds :



```
% medium
m1=mean(U1);
m2=mean(U2);

% intraclass variance matrix
S1= 1/19*(U1-m1)'*(U1-m1);
S2=1/19*(U2-m2)'*(U2-m2);
W=S1+S2;
```

```
% solution
u= W^(-1)*(m1-m2)';

% reprojections
y1= U1*u;
y2= U2*u;
figure, scatter(y1, ones(20,1), 'x');
hold on;
scatter(y2, 2*ones(20,1), 'o');
```

## Conclusions on the LDA

- Benefits :
  - Provides an optimal linear solution to the separation problem
  - If there is an axis, or a linear combination of axes, that leads to good separation, the LDA will tend to select it over the others.
  - The maximized criterion has a statistical meaning
- Disadvantages :
  - Requires inverting $\boldsymbol{W} \Rightarrow$ quickly infeasible if dimension of the order of a thousand.
  - Artificial variables created by linear combination no longer necessarily make sense

# Kernel LDA – derivation

- The kernel trick can be applied to extend ordinary LDA in high dimensional spaces.
- For the 2-class problem, let us denote by $\boldsymbol{x}_i^j$ the $i$-th sample from class $j = 1, 2$, by $l_j$ the number of samples in class $j$, and with abuse of notation, by $\boldsymbol{x}_i$ the $i$-th sample of the whole training set
- The mean of class $j$ in the feature space is

$$\boldsymbol{m}_j^{\phi} = \frac{1}{l_j} \sum_{k=1}^{l_j} \phi(\boldsymbol{x}_k) \tag{5}$$

- Let us also recall equations (3) and (4), which give the Rayleigh ratio and its maximum :

$$J(u) = \frac{\boldsymbol{u}^t \boldsymbol{B} \boldsymbol{u}}{\boldsymbol{u}^t \boldsymbol{W} \boldsymbol{u}}$$
$$\boldsymbol{u} = \boldsymbol{W}^{-1}(\boldsymbol{m}_1 - \boldsymbol{m}_2)$$

# Kernel LDA – derivation

- In the feature space, the Rayleigh ratio now writes :

$$J(\boldsymbol{u}) = \frac{\boldsymbol{u}^t \boldsymbol{B}^\phi \boldsymbol{u}}{\boldsymbol{u}^t \boldsymbol{W}^\phi \boldsymbol{u}}$$

- $\boldsymbol{B}^\phi$ and $\boldsymbol{W}^\phi$ are the between and within-class scatter matrices in the feature space, and follow the same definition as before :

$$\boldsymbol{B}^\phi = (\boldsymbol{m}_2^\phi - \boldsymbol{m}_1^\phi)(\boldsymbol{m}_2^\phi - \boldsymbol{m}_1^\phi)^t$$

$$\boldsymbol{W}^\phi = \sum_{k=1}^{2} \sum_{i=1}^{l_k} (\phi(\boldsymbol{x}_i^k) - \boldsymbol{m}_k^\phi)(\phi(\boldsymbol{x}_i^k) - \boldsymbol{m}_k^\phi)^t$$

- We denote by $k(\boldsymbol{x}, \boldsymbol{y})$ the dot product of the images of any pair of data points $\boldsymbol{x}, \boldsymbol{y}$ in the feature space :

$$k(\boldsymbol{x}, \boldsymbol{y}) =< \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) >= \phi(\boldsymbol{x})^t \phi(\boldsymbol{y}) \qquad (6)$$

# Kernel LDA – derivation

- Using the expansion $\boldsymbol{u} = \sum_{j=1}^{l} \alpha_j \phi(\boldsymbol{x}_j)$, where $l = l_1 + l_2$, and equations (5) and (6), we can evaluate the projection of $\boldsymbol{m}_j^{\phi}$ on $\boldsymbol{u}$ as

$$\boldsymbol{u}^t \boldsymbol{m}_i^{\phi} = \frac{1}{l_i} \sum_{j=1}^{l} \sum_{k=1}^{l_i} \alpha_j k(\mathbf{x}_j, \mathbf{x}_k^i) = \boldsymbol{\alpha}^t \boldsymbol{M}_i \tag{7}$$

where $\boldsymbol{M}$ is a $2 \times l$ matrix defined as

$$(\boldsymbol{M}_i)_j = \frac{1}{l_i} \sum_{k=1}^{l_i} k(\mathbf{x}_j, \mathbf{x}_k^i)$$

- Using (7), the numerator of the Rayleigh ratio $J(\boldsymbol{u})$ can be rewritten as

$$\begin{aligned}\boldsymbol{u}^t \boldsymbol{B}^{\phi} \boldsymbol{u} &= \boldsymbol{u}^t (\mathbf{m}_2^{\phi} - \mathbf{m}_1^{\phi})(\mathbf{m}_2^{\phi} - \mathbf{m}_1^{\phi})^t \boldsymbol{u} \\ &= \boldsymbol{\alpha}^t \boldsymbol{M} \boldsymbol{\alpha}\end{aligned}$$

# Kernel LDA – derivation

- Regarding the denominator, a longer calculation shows that the following equality holds true

  $$\boldsymbol{u}^t \boldsymbol{W}^\phi \boldsymbol{u} = \alpha^t \mathbf{N} \alpha$$

  where

  $$\boldsymbol{N} = \sum_{i=1}^{2} \boldsymbol{K}_i (\boldsymbol{I} - \frac{1}{l_i} \mathbf{1}) \boldsymbol{K}_i^t$$

  and

  $$(\boldsymbol{K}_i)_{a,b} = k(\boldsymbol{x}_a, \boldsymbol{x}_b^i)$$

- Hence, the Rayleigh ratio assumes the form

  $$J(\boldsymbol{u}) = \frac{\boldsymbol{\alpha}^t \boldsymbol{M} \boldsymbol{\alpha}}{\boldsymbol{\alpha}^t \boldsymbol{N} \boldsymbol{\alpha}}$$

# Kernel LDA – derivation

- Its maximization is similar to that involved in ordinary LDA, and amounts to finding the leading eigenvector of $\boldsymbol{N}^{-1}\boldsymbol{M}$.
- Let's call $\boldsymbol{\alpha}^*$ the optimal solution. Then $\boldsymbol{u}^* = \sum_{j=1}^{l} \alpha_j^* \phi(\boldsymbol{x}_j)$, and the projection of any data point $\boldsymbol{y}$ on the optimal $\boldsymbol{u}$ equals to

$$< \boldsymbol{y}, \boldsymbol{u}^* >= \sum_{i=1}^{l} \alpha_i^* k(\boldsymbol{y}, \boldsymbol{x}_i)$$

Regularization :

- Computing $\boldsymbol{N}^{-1}$ is problematic as we estimate $l$-dimensional variance-covariance structures using $l$ samples, so $\boldsymbol{N}$ is ill-conditionned
- The problem is circumvented by regularization : we add a small times $\mu$ the identity matrix to it.
- This makes $\boldsymbol{N}$ invertible for $\mu$ large enough

# Kernel LDA – History

- The extension is due to S. Mika et al : *Fisher discriminant analysis with kernels*, Neural Networks for Signal Processing IX : Proceedings of the 1999 IEEE Signal Processing Society Workshop.
- Permanent link : `https://perso.esiee.fr/~hilairex/AIC-5201A/mika.pdf`
- See also the Wikipedia page `https://en.wikipedia.org/wiki/Kernel_Fisher_discriminant_analysis`

# Kernel LDA – Results

Illustration (source : De-Jiang Luo, A. Liu, *Kernel Fisher Discriminant Analysis [...]*)

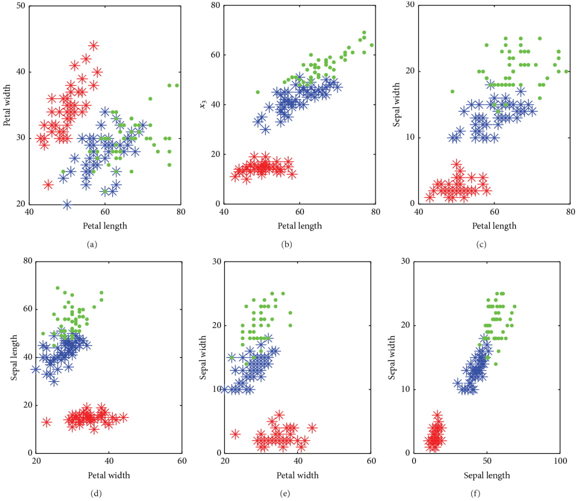
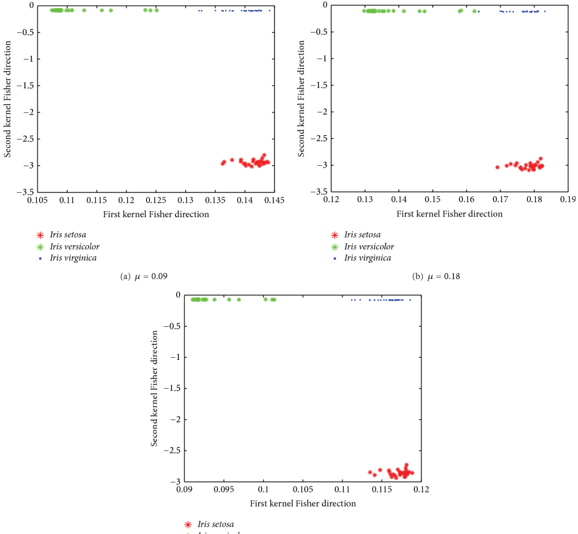


FIGURE 1: Scatter gram plots of *Iris* data: red (C1, *setosa*); green (C2, *virginica*); blue (C3, *versicolor*).

# Kernel LDA – Results

Illustration (source : De-Jiang Luo, A. Liu, *Kernel Fisher Discriminant Analysis [...]*)



(a) $\mu = 0.09$

(b) $\mu = 0.18$

# Kernel LDA – Results

Results from Mika et al.

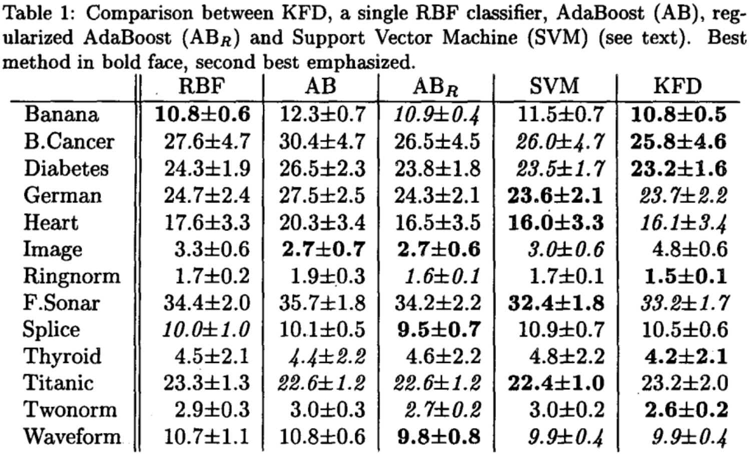


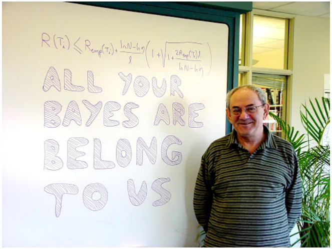
Table 1: Comparison between KFD, a single RBF classifier, AdaBoost (AB), regularized AdaBoost ($AB_R$) and Support Vector Machine (SVM) (see text). Best method in bold face, second best emphasized.

| | RBF | AB | $AB_R$ | SVM | KFD |
|---|---|---|---|---|---|
| Banana | **10.8±0.6** | 12.3±0.7 | *10.9±0.4* | 11.5±0.7 | **10.8±0.5** |
| B.Cancer | 27.6±4.7 | 30.4±4.7 | 26.5±4.5 | *26.0±4.7* | **25.8±4.6** |
| Diabetes | 24.3±1.9 | 26.5±2.3 | 23.8±1.8 | *23.5±1.7* | **23.2±1.6** |
| German | 24.7±2.4 | 27.5±2.5 | 24.3±2.1 | **23.6±2.1** | *23.7±2.2* |
| Heart | 17.6±3.3 | 20.3±3.4 | 16.5±3.5 | **16.0±3.3** | *16.1±3.4* |
| Image | 3.3±0.6 | **2.7±0.7** | **2.7±0.6** | *3.0±0.6* | 4.8±0.6 |
| Ringnorm | 1.7±0.2 | 1.9±0.3 | *1.6±0.1* | 1.7±0.1 | **1.5±0.1** |
| F.Sonar | 34.4±2.0 | 35.7±1.8 | 34.2±2.2 | **32.4±1.8** | *33.2±1.7* |
| Splice | *10.0±1.0* | 10.1±0.5 | **9.5±0.7** | 10.9±0.7 | 10.5±0.6 |
| Thyroid | 4.5±2.1 | *4.4±2.2* | 4.6±2.2 | 4.8±2.2 | **4.2±2.1** |
| Titanic | 23.3±1.3 | *22.6±1.2* | *22.6±1.2* | **22.4±1.0** | 23.2±2.0 |
| Twonorm | 2.9±0.3 | 3.0±0.3 | *2.7±0.2* | 3.0±0.2 | **2.6±0.2** |
| Waveform | 10.7±1.1 | 10.8±0.6 | **9.8±0.8** | *9.9±0.4* | *9.9±0.4* |

# Support vector classification (SVC)

## Origin and motivations :

- Invented by Vladimir Vapnik and his colleagues (at AT&T) during the 90s


$R(T_i) \leqslant R_{emp}(T_i) + \frac{\ln N - \ln \eta}{l}\left(1+\sqrt{1+\frac{2R_{emp}(T_i)l}{\ln N - \ln \eta}}\right)$
ALL YOUR
BAYES ARE
BELONG
TO US

- Motivation : learn the surface that best separates two classes from sample data
- The surface is a hyperplane in the linear case
- The extension to the nonlinear case uses the (*kernel trick*) to classify in a high dimensional space (possibly infinite) without ever having to explicitely compute any representation in that space

The separable linear case

## The separable linear case

- We have data $\boldsymbol{x}_i$, $i = 1, ..., n$ in a vector space $R^d$ of dimension $d$
- These data come from 2 classes $\omega_-$, $\omega_+$, and are labeled by one of the variables $y_i$ that we also know for each sample :

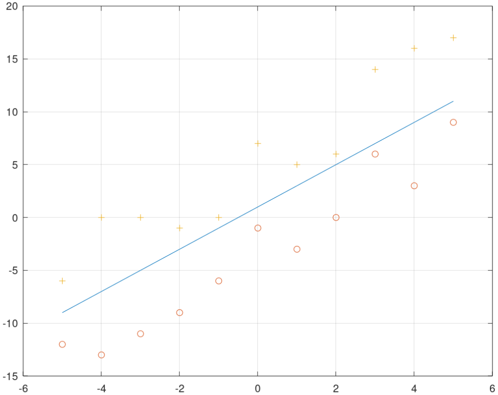
$$y_i = \begin{cases} -1 & \text{si } \boldsymbol{x}_i \in \omega_- \\ +1 & \text{si } \boldsymbol{x}_i \in \omega_+ \end{cases}$$

- These data are *linearly separable* if there exists a hyperplane of $R^d$, with parameters $\boldsymbol{w}$ and $b$, such that $\boldsymbol{w}^t\boldsymbol{x}_i + b$ has the same sign as $y_i$ for all $i$ :

$$(\boldsymbol{w}^t\boldsymbol{x}_i + b)y_i > 0, \quad \forall i = 0, ..., n \tag{8}$$

Equality is reached when $\boldsymbol{x}_i$ lies exactly on the plane. In this case, the point is unclassifiable. We exclude it for now.

Example : the data below is linearly separable by the line $2x - y + 1 = 0$



## Proposition

*If there exists* $\mathbf{w}$ *and* $b$ *satisfying (8), then there exists* $\mathbf{w}'$ *and* $b'$ *satisfying*

$$(\mathbf{w}'^t\mathbf{x}_i + b')y_i \geq 1, \quad \forall i = 0, ..., n \tag{9}$$

*and such that equality is reached at least once.*

This form is called the *canonical form* of the linear separation problem. We shall only use it hereafter :

## Definition

A set of points $\mathbf{x}_i$ labeled by $y_i = \pm 1$ is linearly separable if there exists a hyperplane $(\mathbf{w}, b)$ such that

$$(\mathbf{w}^t\mathbf{x}_i + b)y_i \geq 1, \quad \forall i = 1, ..., n \tag{10}$$

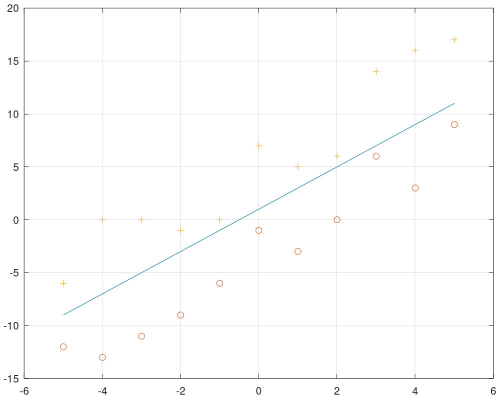
### Definition

The margin $\gamma$ associated with a separating hyperplane is the smallest of the Euclidean distances from this plane to the data points :

$$\gamma = \inf_i \frac{|\boldsymbol{w}^t \boldsymbol{x} + b - \boldsymbol{x}_i|}{||\boldsymbol{w}||} \tag{11}$$

The margin somehow represents the ≪ security ≫ that the linear classifier provides with respect to the supplied data.

Example : on the example of the previous data, the margin is $1/\sqrt{5}$. Note that the distance between the two hyperplanes parallel to the first, and $1/\sqrt{5}$ apart from it, each touches a data point. The distance between these parallel hyperplanes is $2/\sqrt{5}$.



### Lemma

*If the separation problem is put in canonical form (9), then the margin of the separating hyperplane is* $1/||\boldsymbol{w}||$.

Proof.

$$\gamma = \inf_i \frac{|\boldsymbol{w}^t \boldsymbol{x} + b - \boldsymbol{x}_i|}{||\boldsymbol{w}||} = 1/||\boldsymbol{w}||$$

because the equality $|\boldsymbol{w}^t \boldsymbol{x} + b - \boldsymbol{x}_i| = 1$ is reached at least once per hypothesis.
Consequence : the maximum margin separator hyperplane is the one that minimizes $||\boldsymbol{w}||$, or even $||\boldsymbol{w}||^2$ while continuing to maintain $y_i(\boldsymbol{w}^t \boldsymbol{x}_i + b) \geq 1$. Such a hyperplane is thus a solution of the following quadratic program :

$$\begin{aligned} \text{min!}_{\boldsymbol{w},b} \quad & \frac{1}{2}||\boldsymbol{w}||^2 \\ \text{s.c.} \quad & (\boldsymbol{w}^t \boldsymbol{x}_i + b) y_i \geq 1, \forall i = 1, ..., n \end{aligned} \tag{12}$$

Its generalized Lagrangian write

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{n} \alpha_i((\boldsymbol{w}^t \boldsymbol{x}_i + b)y_i - 1) \tag{13}$$

Vanishing its gradient leads to

$$\begin{aligned} \nabla_{\boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i = \boldsymbol{0} \\ \Rightarrow \boldsymbol{w}^* = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \end{aligned} \tag{14}$$

and

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{15}$$

Plugging (14) and (15) into (13), we obtain

$$\begin{aligned} L &= \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{n} \alpha_i (y_i \boldsymbol{w}^t \boldsymbol{x}_i + b) - 1 \\ &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \boldsymbol{x}_i^t \boldsymbol{x}_j - \sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \boldsymbol{x}_i^t \boldsymbol{x}_j + \sum_{i=1}^{n} \alpha_i \\ &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \boldsymbol{x}_i^t \boldsymbol{x}_j \end{aligned}$$

As a result, the dual problem is written

$$\begin{gathered} \text{max!}_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \boldsymbol{x}_i^t \boldsymbol{x}_j \\ \text{s.c.} \sum_{i=1}^{n} \alpha_i = 0 \\ \alpha_i \geq 0, \forall i = 1, ..., n \end{gathered}$$

The equation (14) is very important :

- It shows that the optimal solution to the problem $\boldsymbol{w}^*$ is expressed as a linear combination of the data
- Some of the $\alpha_i$ may be null. If this is the case, this means that the related point $i$ does not participate in the definition of the optimal plane
- Some of the $\alpha_i$ will not be null. The data points associated with them are called support points. They participate in the definition of the optimal plane, and they touch either the lower or the upper hyperplane, according to the sign of $y_i$.

We can summarize the two results in a single equality :

$$\alpha_i(y_i(< \boldsymbol{w}^*, \boldsymbol{x}_i > +b) - 1) = 0, \quad \forall i = 1, ..., n \tag{16}$$

- Equation (15) does not provide the optimal $b$.
- To obtain it, we must reconsider that the equality in the constraints (12) is reached at least twice, once for a $y_i = +1$, the other for $y_i = -1$. Which gives

$$b^* = 1 - \min_{y_i=+1} \boldsymbol{w}^t \boldsymbol{x}_i = 1 + \max_{y_i=-1} \boldsymbol{w}^t \boldsymbol{x}_i \tag{17}$$

- Finally, using (14) and (16), we get :

$$\begin{aligned} ||\boldsymbol{w}^*||^2 &= < \boldsymbol{w}^*, \boldsymbol{w}^* > \\ &= \left( \sum_{j=1}^{n} \alpha_j y_j \boldsymbol{x}_j \right)^t \left( \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \right) \end{aligned}$$

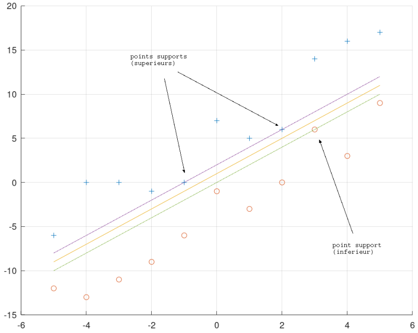
$$= \sum_{j=1}^{n} \alpha_j y_j \sum_{i=1}^{n} \alpha_i < \boldsymbol{x}_j, \boldsymbol{x}_i >$$

$$= \sum_{j=1}^{n} \alpha_j (1 - y_j b^*)$$

$$= \sum_{j=1}^{n} \alpha_j$$

- The expression for the margin in dual space is thus

$$\gamma = \frac{1}{||\boldsymbol{w}||} = \left( \sum_{i=1}^{n} \alpha_i \right)^{-1/2}$$

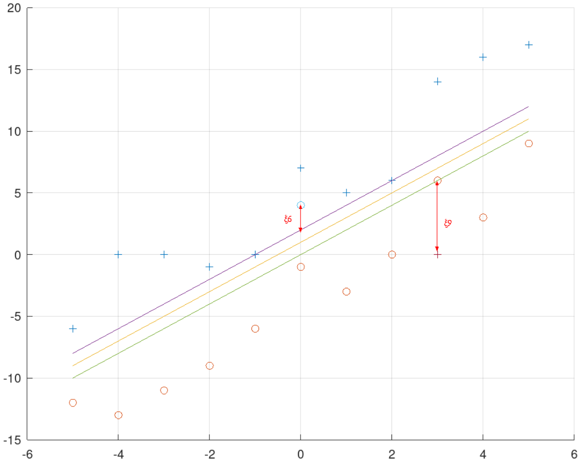


Remark :

- $d$ points in general position are enough to define one of the lower or upper hyperplanes in $R^d$
- 1 single point is enough to define the other
- We will therefore typically observe $d+1$ support points for any data
- But there can be more (points going through the already defined hyperplanes), or less (2 points can be enough to define everything)

The non-separable linear case

- If the data is not separable, the constraints (8) can no longer be maintained and must be relaxed
- We introduce gap variables $\xi_i \geq 0$, and we replace $\geq 1$ by $\geq 1 - \xi_i$



- We must then make a compromise between the total error induced by the $\xi_i$, and the quality of the hyperplane found
- The simplest relaxed version of (12) is

$$\begin{aligned} \min!_{\boldsymbol{w},b} \quad & \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.c.} \quad & (\boldsymbol{w}^t\boldsymbol{x}_i + b)y_i \geq 1 - \xi_i, \forall i = 1, ..., n \\ & \xi_i \geq 0, \quad \forall i = 1, ..., n \end{aligned} \tag{18}$$

  in which $C > 0$ is a constant which weights the importance of the error on the $\xi_i$ in the solution
- We talk about C-SVC for this reason
- We chose $\sum_{i=1}^{n}\xi_i$ , but we could have chosen $\sum_{i=1}^{n}\xi_i^2$, or any increasing function with the $\xi_i$.

- The generalized Lagrangian of (18) becomes

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\xi}) = \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n}\xi_i$$

$$- \sum_{i=1}^{n}\alpha_i\left[(\boldsymbol{w}^t\boldsymbol{x}_i + b)y_i - 1 + \xi_i\right] - \sum_{i=1}^{n}\beta_i\xi_i$$

- Vanishing its partial derivatives yields

$$\nabla_{\boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i = \boldsymbol{0}$$

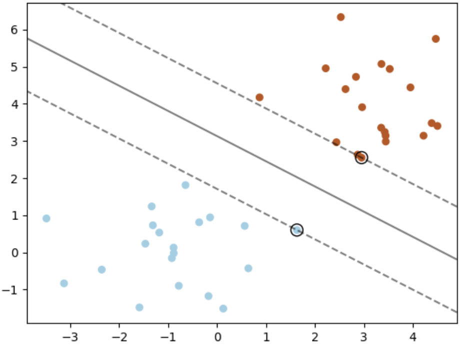
$$\Rightarrow \boldsymbol{w}^* = \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i \tag{19}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n}\alpha_i y_i = 0$$

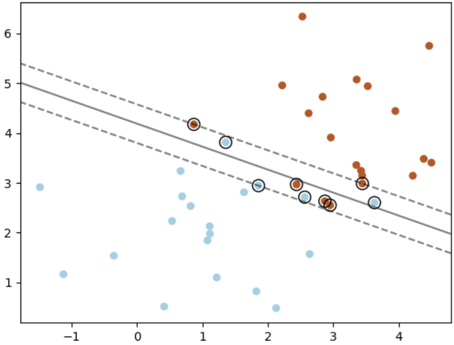
as before, and

$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - \beta_i = 0, \quad \forall i = 1, ..., n \tag{20}$$

- The problem to be solved in the dual space is still linear : the addition of the $\xi_i$ introduces as many new variables, but does not change their nature.
- Comparison between separable and inseparable cases



- Comparison between separable and inseparable cases



- Support points are now those for which $\alpha_i \neq 0$ and $\beta_i \neq 0$
- There can be more than $d + 1$ of them
- An SVC solution is as dubious as the number of support points it involves is high.

# Non-linear support vector classification

# Non-linear SVC

- Recall that the linear cases leads to solving the the problem presented at equation (12) :

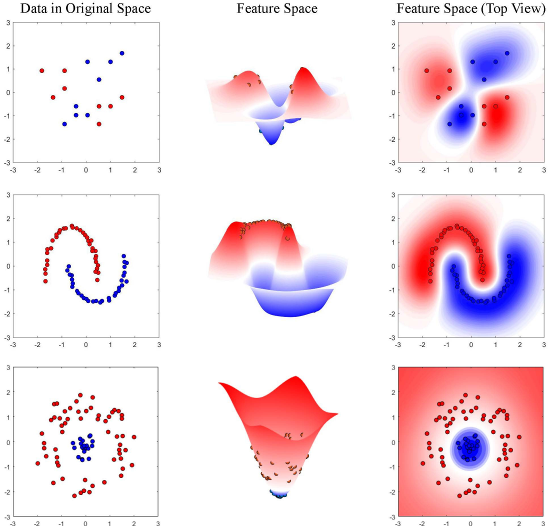
$$\begin{aligned} \text{min!}_{\boldsymbol{w},b} \quad & \frac{1}{2}||\boldsymbol{w}||^2 \\ \text{s.c.} \quad & (\boldsymbol{w}^t\boldsymbol{x}_i + b)y_i \geq 1, \forall i = 1, ..., n \end{aligned}$$

- Suffice to notice that $||\boldsymbol{w}||^2 =< \boldsymbol{w}, \boldsymbol{w} >$ and that $\boldsymbol{w}^t\boldsymbol{x}_i =< \boldsymbol{w}, \boldsymbol{x}_i >$ to see that its resolution remains unchanged :
    - we end up with equations (14) and (17), as we did it for the linear, separable case
    - and with equation (20) in the inseparable case

  both irrespective of the dot product ⇒ Support vector machines are designed to be used with *any* kernel, without any need to modify the method itself.

# Non-linear SVC

Illustration (source : K.E Pilario et al., *A Review of Kernel Methods for Feature Extraction in Nonlinear Process Monitoring*)


Data in Original Space
Feature Space
Feature Space (Top View)