

RBF Networks

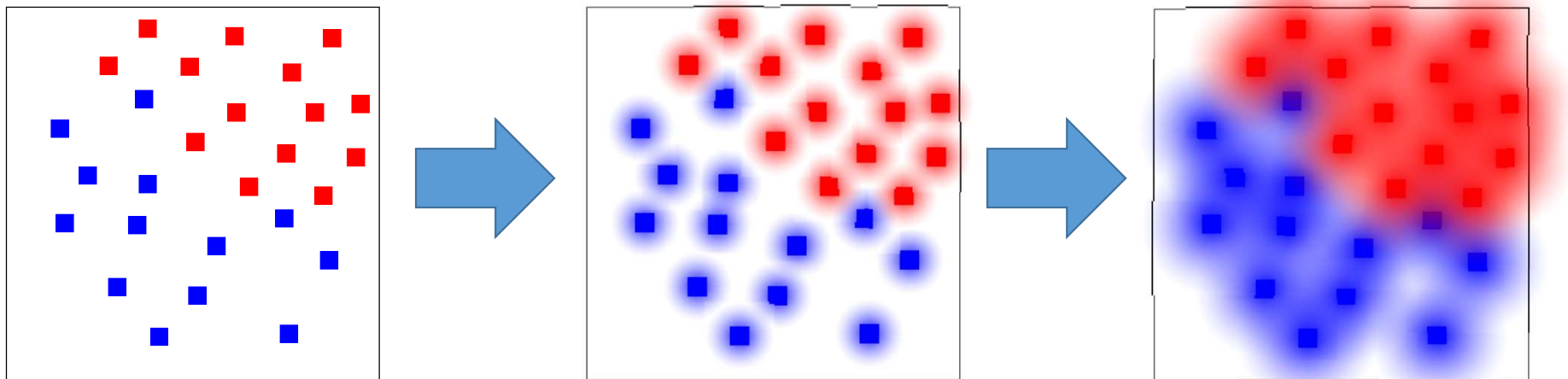
Intuition

Retours sur l'apprentissage « par cœur »

Intuition

Retours sur l'apprentissage « par cœur »

Conserver les exemples et attribuer une 'zone d'influence'



Principes

Régression RBF Naïf :

$$output(x) = \sum_{n=1}^N w_n e^{-\gamma ||X - X_n||^2}$$

Classification RBF Naïf :

$$output(x) = sign(\sum_{n=1}^N w_n e^{-\gamma ||X - X_n||^2})$$

Principes

Régression RBF Naïf :

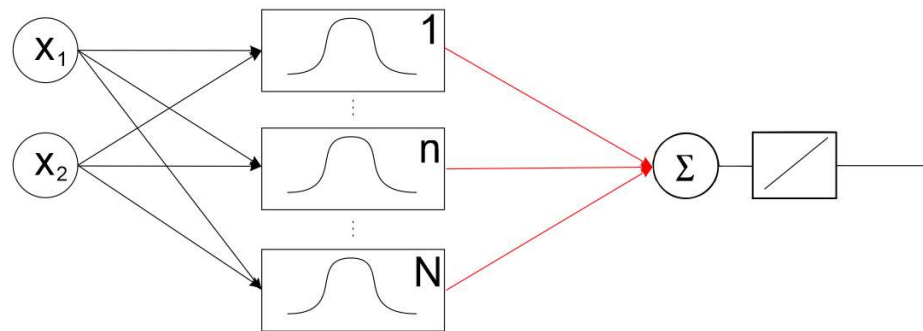
$$output(x) = \sum_{n=1}^N w_n e^{-\gamma ||X - X_n||^2}$$

Classification RBF Naïf :

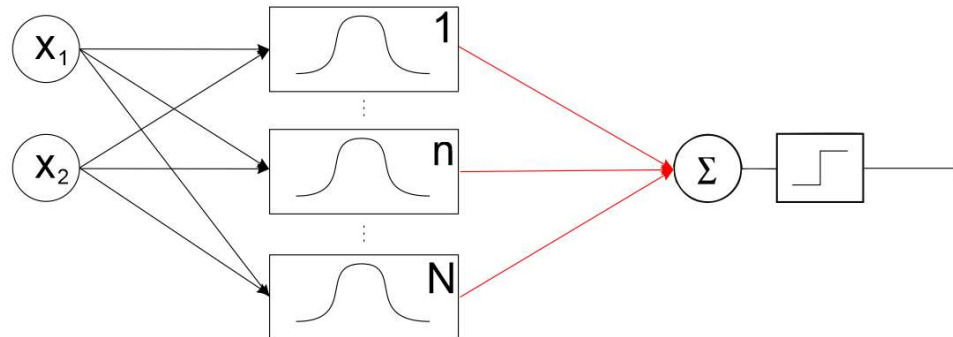
$$output(x) = sign(\sum_{n=1}^N w_n e^{-\gamma ||X - X_n||^2})$$

Principes

Régression RBF Naïf :



Classification RBF Naïf :



Principes :

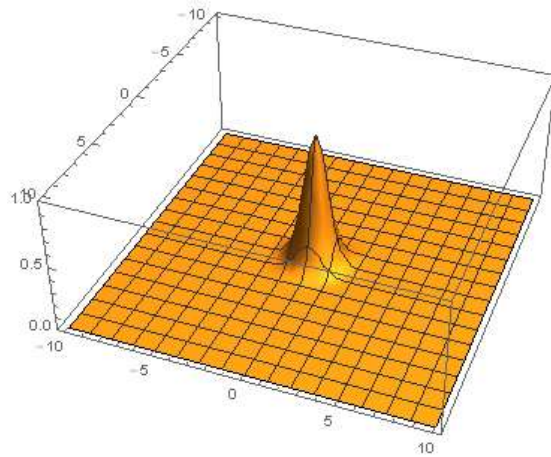
Trouver W pour un RBF naïf :

$$\text{Soit } \phi = \begin{bmatrix} e^{-\gamma \|X_1 - X_1\|^2} & \dots & e^{-\gamma \|X_1 - X_N\|^2} \\ \vdots & \ddots & \vdots \\ e^{-\gamma \|X_N - X_1\|^2} & \dots & e^{-\gamma \|X_N - X_N\|^2} \end{bmatrix} \text{ et } Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_N \end{bmatrix}$$

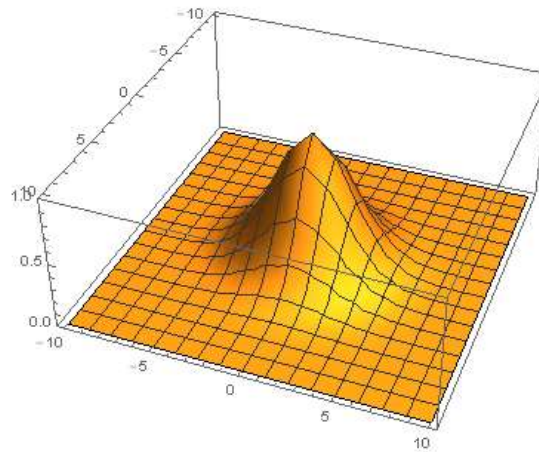
$$\text{Alors } W = \phi^{-1} Y$$

Principes :

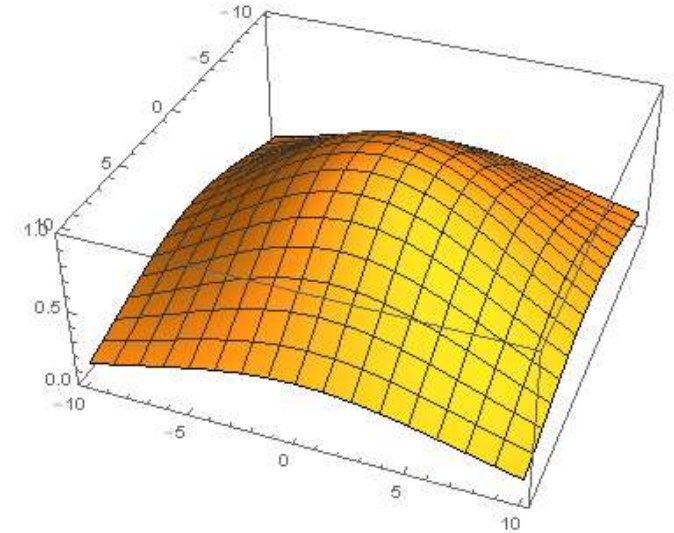
Impact du choix de Gamma :



$$\gamma = 1$$



$$\gamma = 0,1$$



$$\gamma = 0,01$$

Principes

Plus on a d'exemples à disposition, mieux c'est ?

Principes

En a d'exemple disposition, mieux c'est ?



Principes

Plus on a d'exemples à disposition, mieux c'est ?

Nombre de w_i = nombre d'exemples !



Principes

Plus on a d'exemples à disposition, mieux c'est ?

Nombre de w_i = nombre d'exemples !

Mauvais signe pour la généralisation.



Intuition

Ne pas prendre tous les exemples !

Intuition

Ne pas prendre tous les exemples !

Elire des 'représentants'

Principes

k-Means

Méthode exacte : NP-Difficile !

Algorithme de LLoyd

Répéter :

1 :

$$\mu_k = \frac{1}{|S_k|} \sum_{x_n \in S_k} X_n$$

2 :

$$S_k = \{X_n \mid \forall l, \|X_n - \mu_k\| \leq \|X_n - \mu_l\|\}$$

RBF utilisant K centres

Trouver W pour un RBF utilisant K Centres :

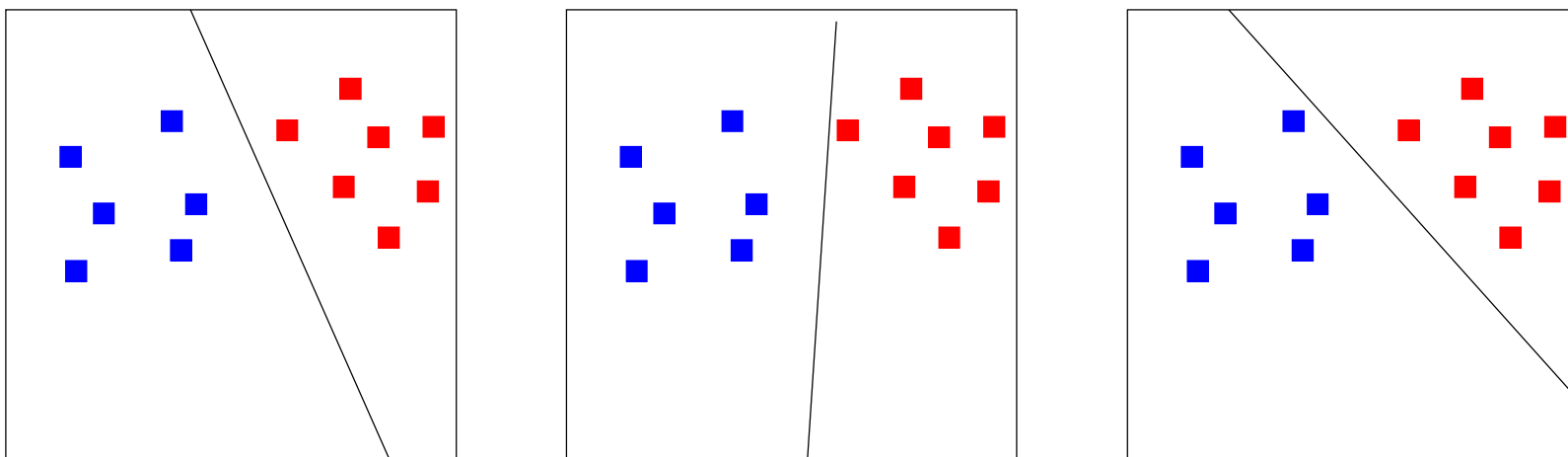
$$\text{Soit } \phi = \begin{bmatrix} e^{-\gamma \|x_1 - \mu_1\|^2} & \dots & e^{-\gamma \|x_1 - \mu_K\|^2} \\ \vdots & \ddots & \vdots \\ e^{-\gamma \|x_N - \mu_1\|^2} & \dots & e^{-\gamma \|x_N - \mu_K\|^2} \end{bmatrix} \text{ et } Y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}$$

$$\text{Alors } W = (\phi^T \phi)^{-1} \phi^T Y$$

SVM

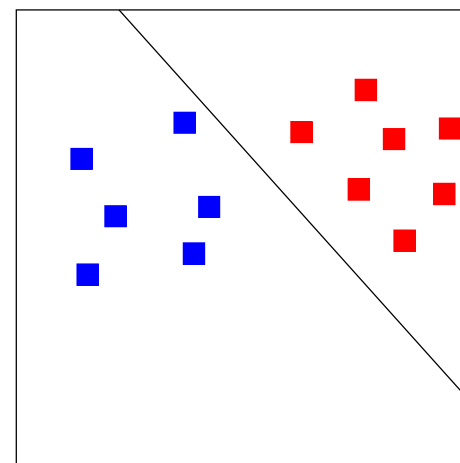
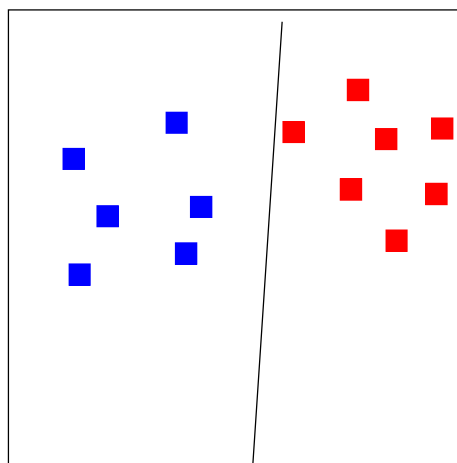
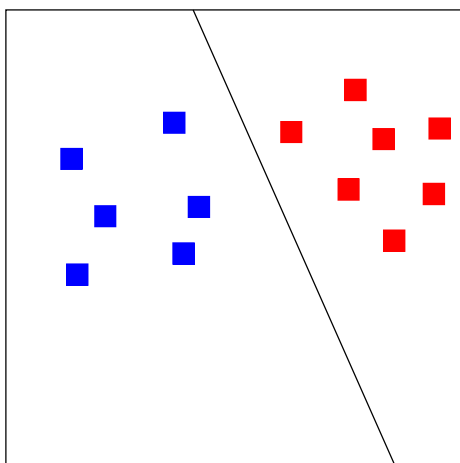
Séparation(s) linéaire(s)

Plusieurs (une infinité) de séparations linéaires possibles ...

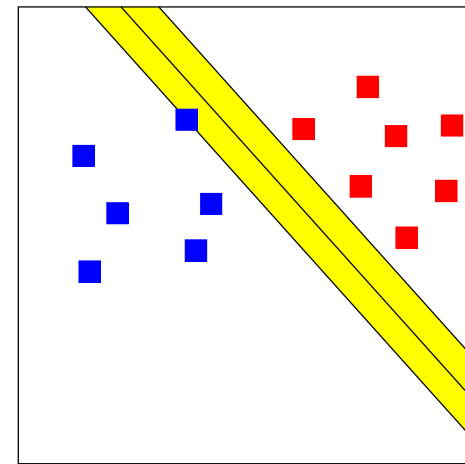
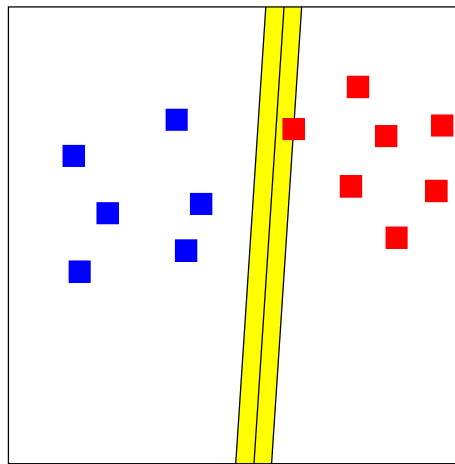
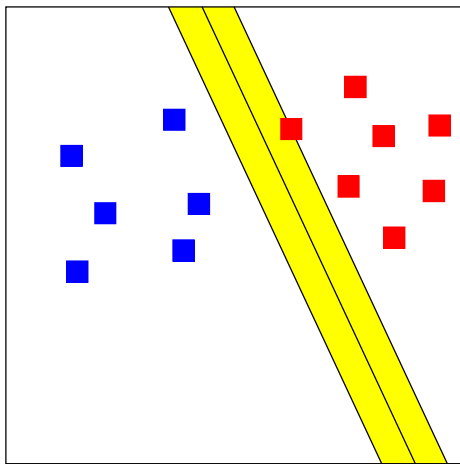


Séparation(s) linéaire(s)

Y en a-t-il une meilleure ?

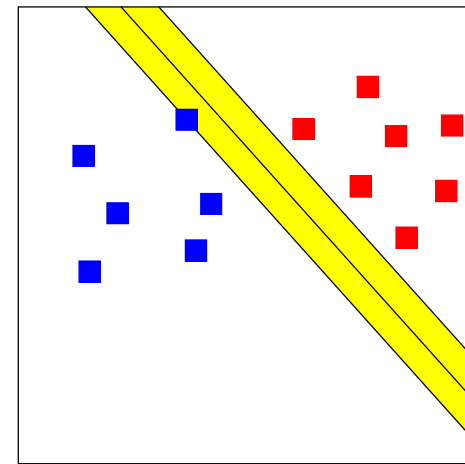
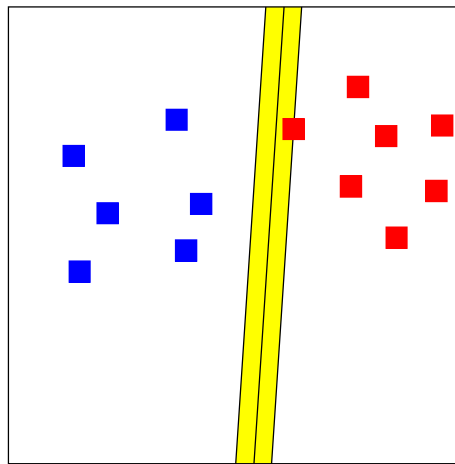
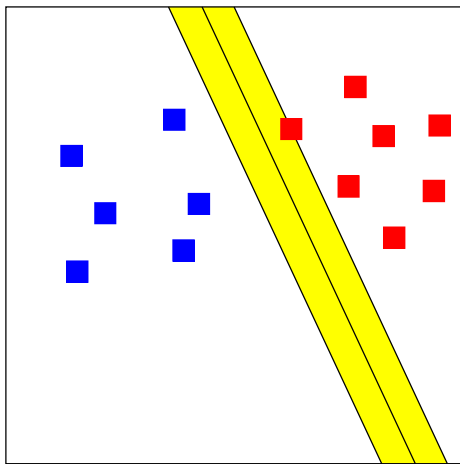


Notion de marge...



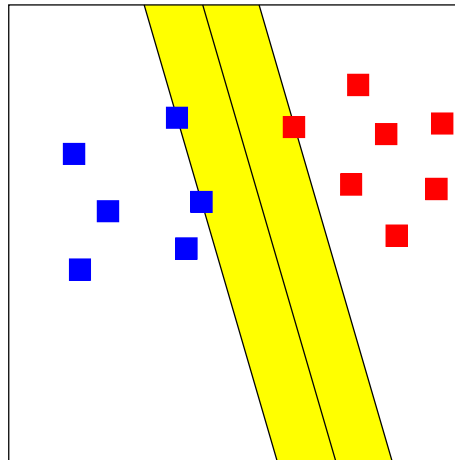
Notion de marge...

Y en a-t-il une meilleure ?



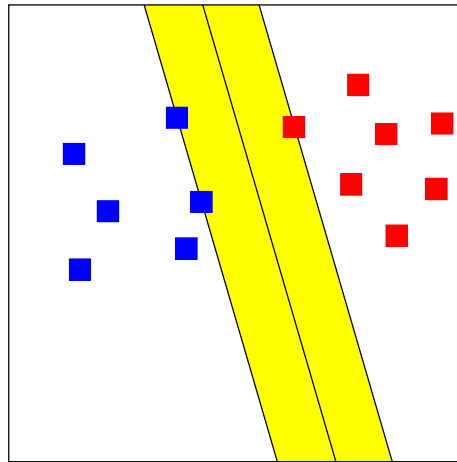
Notion de marge...

Y en a-t-il une meilleure ?



Notion de marge...

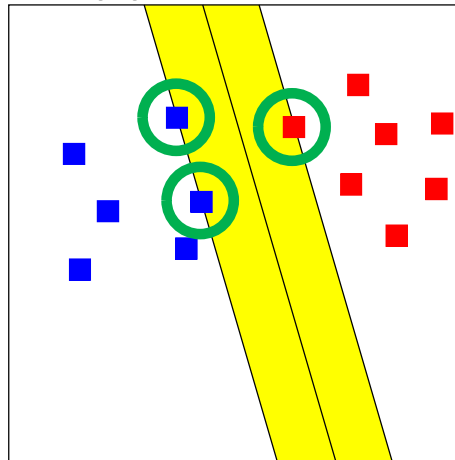
Nouveau problème : trouver les W qui maximisent la marge !



Notion de marge...

Nouveau problème : trouver les W qui maximisent la marge !

C.à.D. : Trouver les vecteurs supports :



Programmation quadratique à la rescousse !

Minimiser :

$$\frac{1}{2} \alpha^T \begin{bmatrix} y_1 y_1 X_1^T X_1 & \cdots & y_1 y_N X_1^T X_N \\ \vdots & \ddots & \vdots \\ y_N y_1 X_N^T X_1 & \cdots & y_N y_N X_N^T X_N \end{bmatrix} \alpha + [-1 \quad \dots \quad -1] \alpha$$

Sous contraintes :

$$Y^T \alpha = 0$$

Avec :

$$\alpha \geq 0$$

Programmation quadratique à la rescousse !

Une fois α obtenu, on peut retrouver W :

$$W = \sum_{n=1}^N \alpha_n y_n X_n$$

Programmation quadratique à la rescousse !

Une fois α obtenu, on peut retrouver W :

$$W = \sum_{n=1}^N \alpha_n y_n X_n$$

Attention, il nous manque w_0 !

Programmation quadratique à la rescousse !

Pour trouver w_0 :

1 – Choisir un X_n tq $\alpha_n > 0$ c.à.d un **Vecteur Support** !

2 – Sachant que $y_n(W^T X_n + w_0) = 1$

3 – Cela nous donne :

$$w_0 = \frac{1}{y_n} - \sum_i w_i X_{ni}$$

Programmation quadratique à la rescousse !

Une fois α obtenu, on peut retrouver W :

$$W = \sum_{n=1}^N \alpha_n y_n X_n$$

N exemples \Rightarrow N paramètres ?

Programmation quadratique à la rescousse !

Une fois α obtenu, on peut retrouver W :

$$W = \sum_{n=1}^N \alpha_n y_n X_n$$

N exemples \Rightarrow N paramètres ?

Si X_i n'est pas un vecteur support, alors $\alpha_i = 0$!

Programmation quadratique à la rescousse !

Une fois α obtenu, on peut retrouver W :

$$W = \sum_{n=1}^N \alpha_n y_n X_n$$

N exemples \Rightarrow N paramètres ?

Si X_i n'est pas un vecteur support, alors $\alpha_i = 0$!

Ainsi, nous avons autant de paramètres dans notre modèle que de vecteur support \Rightarrow bonne généralisation !

Machine à noyaux

Retour sur les SVMs

Si nos exemples sont de grande dimension,

$$\begin{bmatrix} y_1 y_1 X_1^T X_1 & \cdots & y_1 y_N X_1^T X_N \\ \vdots & \ddots & \vdots \\ y_N y_1 X_N^T X_1 & \cdots & y_N y_N X_N^T X_N \end{bmatrix}$$

Sera difficile à calculer !

Retour sur les SVMs

Projection des entrées dans un autre espace (le retour) :

$$\begin{bmatrix} y_1 y_1 \mathbf{z}_1^T \mathbf{z}_1 & \cdots & y_1 y_N \mathbf{z}_1^T \mathbf{z}_N \\ \vdots & \ddots & \vdots \\ y_N y_1 \mathbf{z}_N^T \mathbf{z}_1 & \cdots & y_N y_N \mathbf{z}_N^T \mathbf{z}_N \end{bmatrix}$$

Si l'espace est de dimension supérieure à l'espace de départ, cela devrait être encore pire !

Retour su

Projection des

Si l'espace est
devrait être en



art, cela

Retour sur les SVMs

Projection des entrées dans un autre espace (le retour) :

$$\begin{bmatrix} y_1 y_1 \mathbf{z}_1^T \mathbf{z}_1 & \cdots & y_1 y_N \mathbf{z}_1^T \mathbf{z}_N \\ \vdots & \ddots & \vdots \\ y_N y_1 \mathbf{z}_N^T \mathbf{z}_1 & \cdots & y_N y_N \mathbf{z}_N^T \mathbf{z}_N \end{bmatrix}$$

Si l'espace est de dimension supérieure à l'espace de départ, cela devrait être encore pire !

Cela dépend du type de transformation !

Retour sur les SVMs

Cela dépend du type de transformation !

Nous n'avons besoin que de l'existence de la possibilité d'effectuer produit scalaire dans le nouvel espace !

$$\begin{bmatrix} y_1 y_1 K(X_1, X_1) & \cdots & y_1 y_N K(X_1, X_N) \\ \vdots & \ddots & \vdots \\ y_N y_1 K(X_N, X_1) & \cdots & y_N y_N K(X_N, X_N) \end{bmatrix}$$

Différents noyaux :

Noyau Polynomial de degré Q :

$$K(x_n, x_m) = (1 + x_n^T x_m)^Q$$

Noyau à Base Radiale :

$$K(x_n, x_m) = e^{-x_n^2} e^{-x_m^2} e^{2x_n x_m}$$

Equivalent à une
projection dans un
espace de
dimension infinie !

Noyaux :

Degré Q :

$$K(x_n, x_m) = (1 + x_n^T x_m)^Q$$

Sans augmentation
du nombre de
paramètres !

Noyau à Base Radiale :

$$K(x_n, x_m) = e^{-x_n^2} e^{-x_m^2} e^{2x_n x_m}$$