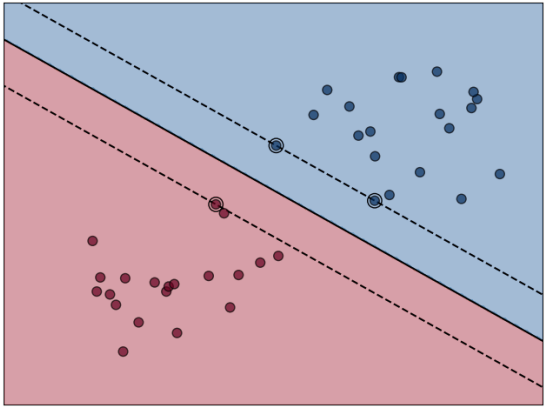
Théorie et Algorithmes de l'Apprentissage Automatique

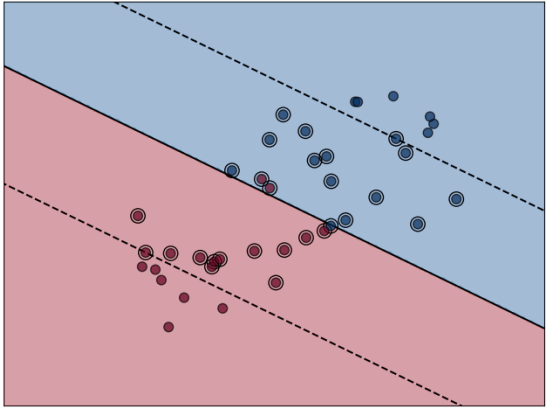
4 - SVM à noyaux

Simon BERNARD
simon.bernard@univ-rouen.fr

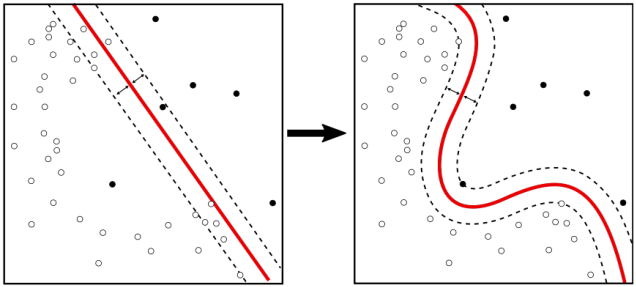
- Problème linéairement séparable
- Toutes les données d'apprentissage sont correctement classées par le séparateur linéaire



- Problème linéairement non-séparable
- Quelques erreurs d'apprentissage : compromis entre marge et nombres d'erreurs

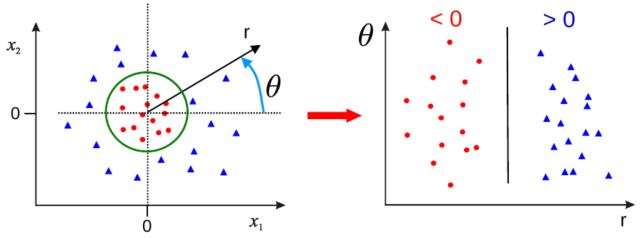


- Les données ne sont pas toujours des vecteurs (textes, graphes, images, etc.)
- Les données peuvent être séparables mais non-linéairement



Transformer un séparateur linéaire en séparateur non-linéaire ?

- Approches d'optimisation non-linéaires : complexité calculatoire trop importante
- Méthodes linéaires bien connues et plus simples
- Gardons les approches linéaires, et travaillons sur les données.
- Exemples : coordonnées polaires



- Dans l'exemple précédent, on a opéré à une transformation non-linéaire des données :

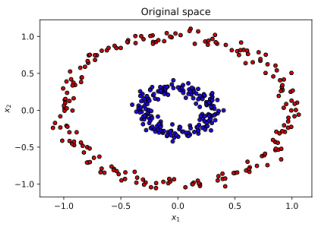
$$\phi(\mathbf{x}) : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \rightarrow \begin{pmatrix} r \\ \theta \end{pmatrix}$$

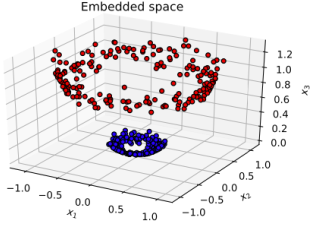
- Les données sont linéairement séparables dans le nouvel espace
- Problème :
  - Fonctionne ici parce que les données s'y prêtent.
  - Comment faire sans *a priori* sur les données ?
  - Comment faire quand les données ne sont pas des vecteurs ?
- Solution : l'astuce du noyau (*kernel trick*)

Astuce du noyaux

Objectif :

- Trouver une transformation ϕ vers un espace $\mathcal{F}$ plus riche (*embedding*)
- Entraîner un SVM dans l'espace résultant, i.e. avec les $\{\phi(\mathbf{x}_i), y_i\}$





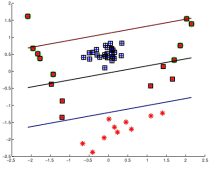
$$\phi(\mathbf{x}) : \begin{pmatrix} x_{(1)} \\ x_{(2)} \end{pmatrix} \rightarrow \begin{pmatrix} x_{(1)}^2 \\ x_{(2)}^2 \\ x_{(1)}^2 + x_{(2)}^2 \end{pmatrix}$$

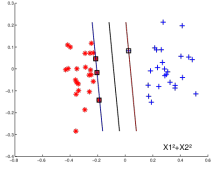
- SVM *soft-margin* dans $\mathcal{F}$:

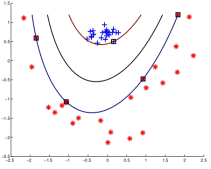
$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.c.} \quad & y_i\left(\mathbf{w}^\top\phi(\mathbf{x}_i) + b\right) \geq 1 - \xi_i, && i = 1,\dots,n \\ & \xi_i \geq 0, && i = 1,\dots,n \end{aligned}$$

  avec $\mathbf{w} \in \mathcal{F}$

- Le modèle $h(\mathbf{x}) = \mathbf{w}^\top\phi(\mathbf{x}) + b$ est non-linéaire dans l'espace orginal







- Le problème dual devient :

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \phi(\mathbf{x}_j)^\top \phi(\mathbf{x}_i) \\ \text{s.c.} \quad & 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned}$$

- La fonction de décision devient :

$$h(\mathbf{x}) = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \mathbf{x}_i^\top \mathbf{x} + b \quad \Rightarrow \quad h(\mathbf{x}) = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}) + b$$

- Dans cette fonction de décision :

$$h(\mathbf{x}) = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}) + b$$

  $\phi(\mathbf{x})$ n'intervient que sous la forme d'un produit scalaire $\phi(\mathbf{x}_i)^\top \phi(\mathbf{x})$
- Nous n'avons pas besoin de calculer $\mathbf{w}$, ni de connaître $\phi(\mathbf{x})$ explicitement
- Il suffit de connaître la fonction k telle que :

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$$

- Conclusion : simplement calculer le produit scalaire $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{F}}$, revient à faire fonctionner le SVM linéaire dans un espace $\mathcal{F}$ implicite (*embedding*)

- La fonction k est appelée fonction noyau :

$$k(.,.) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

- Un SVM à noyau est défini par le problème dual :

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.c.} \quad & 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned}$$

- et la fonction de décision :

$$h(\mathbf{x}) = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) + b$$

Qu’est-ce qu’un noyau ?

Noyau défini positif

Un noyau (*kernel*) k est une fonction $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$:

$$k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle_{\mathcal{F}}$$

avec $\mathbf{x}, \mathbf{z} \in \mathcal{X}$ et $\mathcal{F}$ un espace de caractéristiques muni d'un produit scalaire $\langle \cdot, \cdot \rangle_{\mathcal{F}}$

- Pour les SVM : noyau défini positif
- Pourquoi ? pour garantir que le problème dual des SVM reste bien posé

Noyau défini positif

Un noyau $k(\mathbf{x}, \mathbf{z}) \in \mathcal{X} \times \mathcal{X}$ est dit *défini positif* si :

- il est symétrique : $\forall \mathbf{x}, \mathbf{z} \in \mathcal{X}, \quad k(\mathbf{x}, \mathbf{z}) = k(\mathbf{z}, \mathbf{x})$
- $\forall n \in \mathbb{N}, \forall \{\mathbf{x}_i\}_{i=1,n} \in \mathcal{X}, \forall \{\alpha_i\}_{i=1,n} \in \mathbb{R}$:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0$$

Il est dit strictement défini positif si pour $\alpha_i \neq 0$:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) > 0$$

Fonction noyau ≈ mesure de similarité

- Intuitivement, un noyau est une mesure de similarité :
  - $\forall \mathbf{x}, \mathbf{z} \in \mathcal{X}, \quad k(\mathbf{x}, \mathbf{z}) \geq 0$ (positivité)
  - $\forall \mathbf{x}, \mathbf{z} \in \mathcal{X}, \quad k(\mathbf{x}, \mathbf{z}) = k(\mathbf{z}, \mathbf{x})$ (symétrie)
  - $\forall \mathbf{x}, \mathbf{z} \in \mathcal{X}, \mathbf{z} \neq \mathbf{x}, \quad k(\mathbf{x}, \mathbf{z}) > k(\mathbf{x}, \mathbf{x})$ (uniformité)
  - $\forall \mathbf{x}, \mathbf{z} \in \mathcal{X}, \quad k(\mathbf{x}, \mathbf{z}) = k(\mathbf{x}, \mathbf{x}) \Leftrightarrow \mathbf{x} = \mathbf{z}$ (identité)
- Au lieu de travailler dans un espace de représentation, l'apprentissage se base sur des similarités entre les données
- Permet d'appliquer le principe de non-linéarisation même si $\mathcal{X} \not\subset \mathbb{R}^d$:

$$\mathbf{k}\left(\quad , \quad \right)$$

- De façon équivalente, on peut montrer qu'un noyau k est défini positif si $\forall n \in \mathbb{N}, \forall \{\mathbf{x}_i\}_{i=1,n} \in \mathcal{X}$ la matrice $\mathbf{K}$, appelée matrice de Gram, définie comme :

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \dots & k(\mathbf{x}_1, \mathbf{x}_n) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \dots & k(\mathbf{x}_2, \mathbf{x}_n) \\ \dots & \dots & \dots & \dots \\ k(\mathbf{x}_n, \mathbf{x}_1) & k(\mathbf{x}_n, \mathbf{x}_2) & \dots & k(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix}$$

  est définie positive

- Une matrice $\mathbf{M}$ est définie positive si

$$\mathbf{z}^\top \mathbf{M}\mathbf{z} > 0, \quad \forall \mathbf{z} \in \mathbb{R}^n \neq 0$$

- Les méthodes à noyaux sont des méthodes qui prennent ce type de matrices en entrée

Exemples de noyaux définis positifs

$$k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^\top \mathbf{z}$$

- $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$
- Symétrique : $\mathbf{x}^\top \mathbf{z} = \mathbf{z}^\top \mathbf{x}$
- Positif :

$$\begin{aligned}
\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) &= \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \mathbf{x}_i^\top \mathbf{x}_j \\
&= \left(\sum_{i=1}^{n} \alpha_i \mathbf{x}_i \right)^\top \left(\sum_{j=1}^{n} \alpha_j \mathbf{x}_j \right) \\
&= \left\| \sum_{i=1}^{n} \alpha_i \mathbf{x}_i \right\|^2 \geq 0
\end{aligned}$$

$$k(\mathbf{x}, \mathbf{z}) = g(\mathbf{x})g(\mathbf{z})$$

- $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$
- $g : \mathcal{X} \to \mathbb{R}$
- Symétrique par construction
- Positif :

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j k(\mathbf{x}_i, \mathbf{x}_j) &= \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j g(\mathbf{x}_i)g(\mathbf{x}_j) \\ &= \left(\sum_{i=1}^{n}\alpha_i g(\mathbf{x}_i)\right)\left(\sum_{j=1}^{n}\alpha_j g(\mathbf{x}_j)\right) \\ &= \left(\sum_{i=1}^{n}\alpha_i g(\mathbf{x}_i)\right)^2 \geq 0\end{aligned}$$

Supposons k_1 et k_2 deux noyaux définis positifs. Alors les noyaux suivants sont également définis positifs :

$k(x, z) = k_1(x, z) + k_2(x, z)$

- $k(x, z) = k_1(x, z) + k_2(x, z) = k_1(z, x) + k_2(z, x) = k(z, x)$
- $\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k_1(x_i, x_j) + \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k_2(x_i, x_j) \geq 0$

$k(x, z) = k_1(x, z) k_2(x, z)$

- $k(x, z) = k_1(x, z) k_2(x, z) = k_1(z, x) k_2(z, x) = k(z, x)$
- $\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) = \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k_1(x_i, x_j) \right) \left(sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k_2(x_i, x_j) \right) \geq 0$

$k(x, z) = \exp(k_1(x, z))$

- $k(x, z) = \exp(k_1(x, z)) = \exp(k_1(z, x)) = k(z, x)$
- $\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \exp(k_1(x, z)) \geq 0$

$$k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^\top \mathbf{z} + c)^p$$

- Cas particulier du noyau quadratique pour $p = 2$ et $c = 1$:

$$k(\mathbf{x}, \mathbf{z}) = \left(\sum_{i=1}^{n} x_i z_i + 1 \right)^2 = \left(\mathbf{x}^\top \mathbf{z} + 1 \right)^2 = 1 + 2\mathbf{x}^\top \mathbf{z} + (\mathbf{x}^\top \mathbf{z})^2$$

- La fonction de projection ϕ s'exprime comme :

$$\phi(\mathbf{x}) = (c, \sqrt{2}cx_1, \ldots, \sqrt{2}cx_n, \ldots, \sqrt{2}x_i x_j, \ldots, x_1^2, \ldots, x_n^2)$$

(mais nous n'avons pas besoin de calculer ces coordonnées, uniquement le noyau)

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}\|^2}{2\sigma^2}\right)$$
$$= \exp\left(-\gamma\|\mathbf{x} - \mathbf{z}\|^2\right)$$

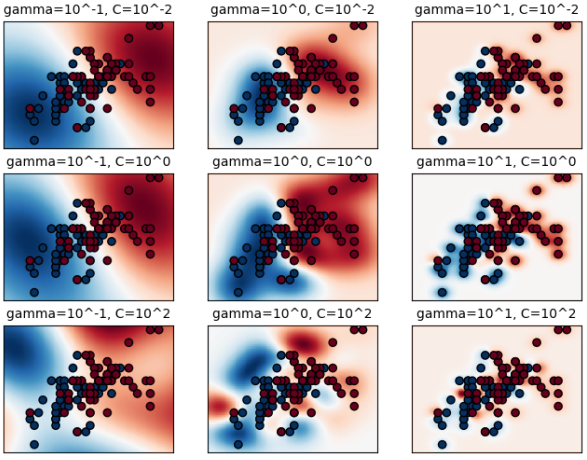
où σ ou γ sont des hyper-paramètres

- Quand $\sigma = 1$:

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{1}{2}\|\mathbf{x} - \mathbf{z}\|^2\right)$$
$$\phi(\mathbf{x}) = \left(\frac{\exp\left(\frac{\|\mathbf{x}\|^2}{2}\right)}{\sqrt{j!}^{1/j}} \binom{j}{n_1, \ldots, n_k}^{1/2} x_{(1)}^{n_1} \ldots x_{(d)}^{n_d}\right)_{j=0,\ldots,\infty, \sum_{i=1}^{j} n_i = j}$$

- σ/γ contrôlent la région d'influence du noyau

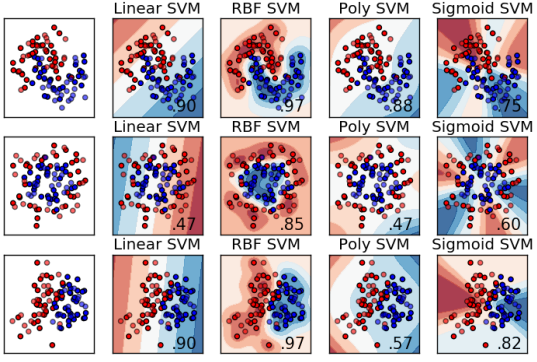
Les performances des SVM avec noyau RBF sont très sensibles à la paramétrisation [1]



1. https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

En pratique

Si possible, choisir en fonction de la géométrie de problème



Sinon...

- Quand la dimension de l'espace de description initial est grande : noyau linéaire
- Sans connaissance à priori sur le problème : noyau RBF

Si k_1 et k_2 sont des noyaux définis positifs sur $\mathcal{X}$ alors les noyaux suivants le sont aussi :

- $k(\mathbf{x}, \mathbf{z}) = \alpha k_1(\mathbf{x}, \mathbf{z})$ avec $\alpha > 0$
- $k(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z}) + k_2(\mathbf{x}, \mathbf{z})$
- $k(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z}) \times k_2(\mathbf{x}, \mathbf{z})$
- $k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^\top \mathbf{B}\mathbf{z}$ avec $\mathbf{B}$ une matrice $n \times n$ semi-définie positive
- $k(\mathbf{x}, \mathbf{z}) = f(\mathbf{x})f(\mathbf{z})$ avec $f()$ une fonction de l'espace des caractéristiques dans $\mathbb{R}$

Il existe des méthodes plus complexes pour construire des noyaux à partir de noyaux existants (par exemple par des convolutions etc...)

Conditions nécessaires :

- Symmétrie : $k(\mathbf{x}, \mathbf{z}) = k(\mathbf{z}, \mathbf{x})$
- Inégalité de Cauchy-Schwarz : $k(\mathbf{x}, \mathbf{z})^2 \leq k(\mathbf{x}, \mathbf{x})k(\mathbf{z}, \mathbf{z})$

Noyau et matrice définie positive :

- Soit $\{\mathbf{x}_i\}_{i=1,n}$ un ensemble de points dans $\mathcal{X}$
- $k(\mathbf{x}, \mathbf{z})$ est un noyau défini positif si la matrice $K = \{k(\mathbf{x}_i, \mathbf{x}_j)\}$ est une matrice symmétrique définie positive
- C'est le cas si $\mathbf{x}^\top \mathbf{K}\mathbf{x} > 0$, avec $\mathbf{x} \neq 0$
- C'est le cas si toutes les valeurs propres de K sont positives

Fernandez-Delgado et al., "Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?", Journal of Machine Learning Research, 2014

- Comparaison de méthodes d'apprentissage : 179 classifieurs et 121 bases de données publiques

Rank	Acc.	κ	Classifier
32.9	82.0	63.5	parRF_t (RF)
33.1	**82.3**	**63.6**	rf_t (RF)
36.8	81.8	62.2	svm_C (SVM)
38.0	81.2	60.1	svmPoly_t (SVM)
39.4	81.9	62.5	rforest_R (RF)
39.6	82.0	62.0	elm_kernel_m (NNET)
40.3	81.4	61.1	svmRadialCost_t (SVM)
42.5	81.0	60.0	svmRadial_t (SVM)
42.9	80.6	61.0	C5.0_t (BST)
44.1	79.4	60.5	avNNet_t (NNET)
45.5	79.5	61.0	nnet_t (NNET)
47.0	78.7	59.4	pcaNNet_t (NNET)
47.1	80.8	53.0	BG_LibSVM_w (BAG)

"The classifiers most likely to be the bests are the random forest (RF) versions [...]. However, the difference is not statistically significant with the second best, the SVM with Gaussian kernel [...]"